

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

REMark®

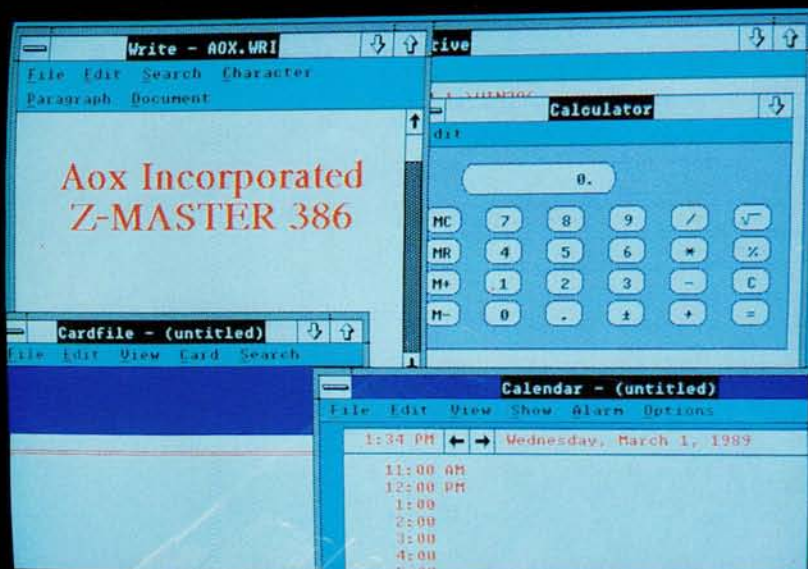
June 1989

**Getting Started With
Batch Files**

Page 19

**A Look at the Plus
in MS-DOS 3.3+**

Page 21

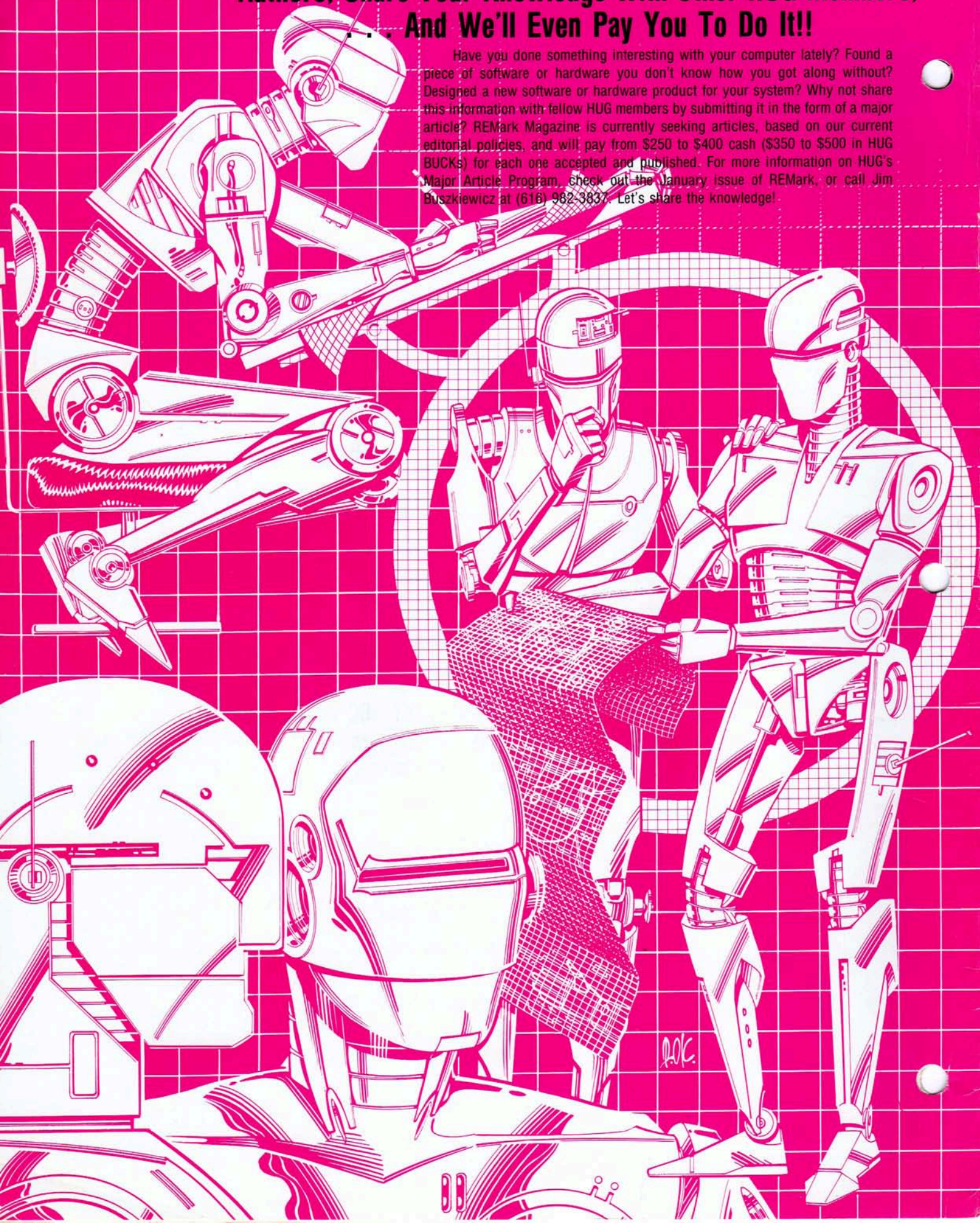


Home Finance Systems

Page 27

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!



REMark®

Volume 10, Issue 6 • June 1989

On The Cover

The AOX Z-Master: Another '386 upgrade kit for your Zenith computer. For a complete review, see the article by Pat Swayne on Page 7.

Resources

HUG Price List	2
Buggin' HUG	5
Classified Ads	42
H/Z Related Products	61
Bug Zapping	63

Reader Service No.		Page No.
101	American Cryptronics, Inc.	37
196	Covox, Inc.	13
197	DMA Technologies	62
198	Enable	62
104	FBE Research Co.	46
105	First Capitol Computer	56
***	HEPCAT	4
137	Jay Gold Software	40
136	Lindley Systems	37
117	Payload Computer Services	30
130	Quikdata, Inc.	38
121	Scottie Systems	61
195	Strategic Alliance, Inc.	62
194	Weltec	42

PC Compatibles

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

PC Compatible

All Speed and No Fluff

Pat Swayne 7

Powering Up

William M. Adney 31

Getting Started With . . . Microsoft Word

Ralph E. Camp 47

H/Z-100 and PC Compatible

Beginner's Batches

Mark Haverstock 19

On the Leading Edge

William M. Adney 21

Home Financial Management

Donald R. Leitch 27

H/Z-100 Only (Not PC)

A Z-100 Assembly Language Program to Send Setup Codes to the C.I.TOH Prowriter Parallel Printer

Les Landers 39

Z-100 Survival Kit #5

Paul F. Herman 51

H/Z-89 and H/Z-100

Dress Up the CP/M 'A>' Prompt with a Picture

Steven W. Vagts 16

General

Programmer's Learning Corner

Don Keller 11

MailMerge Rosters and Mailing Lists — Part II

Ralph E. Camp 43

Graphics Printer or Epson FX? — Part I

John A. Day 57

Managing Editor Jim Buszkiewicz
(616) 982-3837

Software Engineer Pat Swayne
(616) 982-3463

Production Coordinator Lori Lerch
(616) 982-3794

Secretary Margaret Bacon
(616) 982-3463

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

HUG Parts Ordering (616) 982-3463

Contributing Editor William M. Adney

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

Printer Imperial Printing
St. Joseph, MI

	U.S. Domestic	APO/FPO & All Others
Initial	\$22.95	\$37.95*
Renewal	\$19.95	\$32.95*

*U.S. Funds

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

Send Payment to: Heath/Zenith Users' Group
P.O. Box 217
Benton Harbor, MI 49022
(616) 982-3838

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

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

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

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

Copyright © 1989, Heath/Zenith Users' Group

HUG

PRODUCT NAME	PART NUMBER	OPERATING		DESCRIPTION	PRICE
		SYSTEM	SYSTEM		
H8 - H/Z-89/90					
ACCOUNTING SYSTEM	885-8047-37	CPM		BUSINESS	20.00
ACTION GAMES	885-1220-37	CPM		GAME	20.00
ADVENTURE	885-1010	HDOS		GAME	10.00
ASCIRITY	885-1238-37	CPM		AMATEUR RADIO	20.00
AUTOFIL (Z80 ONLY)	885-1110	HDOS		DBMS	30.00
BHBASIC SUPPORT PACKAGE	885-1119-37	HDOS		UTILITY	20.00
CASTLE	885-8032-37	HDOS		ENTERTAINMENT	20.00
CHEAPCALC	885-1131-37	HDOS		SPREADSHEET	20.00
CHECKOFF	885-8010	HDOS		CHECKBOOK SOFTWARE	25.00
DEVICE DRIVERS	885-1105	HDOS		UTILITY	20.00
DISK UTILITIES	885-1213-37	CPM		UTILITY	20.00
DUNGEONS & DRAGONS	885-1093-37	HDOS		GAME	20.00
FLOATING POINT PACKAGE	885-1063	HDOS		UTILITY	18.00
GALACTIC WARRIORS	885-8009-37	HDOS		GAME	20.00
GALACTIC WARRIORS	885-8009-37	CPM		GAME	20.00
GAMES 1	885-1029-37	HDOS		GAMES	18.00
HARD SECTOR SUPPORT PACKAGE	885-1121	HDOS		UTILITY	30.00
HDOS PROGRAMMERS HELPER	885-8017	HDOS		UTILITY	16.00
HOME FINANCE	885-1070	HDOS		BUSINESS	18.00
HUG DISK DUPLICATION UTILITIES	885-1217-37	CPM		UTILITY	20.00
HUG SOFTWARE CATALOG	885-4500	VARIOUS		PRODUCTS THRU 1982	9.75
HUGMAN & MOVIE ANIMATION	885-1124	HDOS		ENTERTAINMENT	20.00
INFO. SYSTEM AND TEL. & MAIL SYSTEM	885-1108-37	HDOS		DBMS	30.00
LOGBOOK	885-1107-37	HDOS		AMATEUR RADIO	30.00
MAGBASE	885-1249-37	CPM		MAGAZINE DATABASE	25.00
MAPLE	885-8005	HDOS		COMMUNICATION	35.00
MAPLE	885-8012-37	CPM		COMMUNICATION	35.00
MICRONET CONNECTION	885-1122-37	HDOS		COMMUNICATION	16.00
MISCELLANEOUS UTILITIES	885-1089-37	HDOS		UTILITY	20.00
MORSE CODE TRANSCIVER	885-8016	HDOS		AMATEUR RADIO	20.00
MORSE CODE TRANSCIVER	885-8031-37	CPM		AMATEUR RADIO	20.00
PAGE EDITOR	885-1079-37	HDOS		UTILITY	25.00
PROGRAMS FOR PRINTERS	885-1082	HDOS		UTILITY	20.00
REMARK VOL 1 ISSUES 1-13	885-4001	N/A		1978 TO DECEMBER 1980	20.00
RUNOFF	885-1025	HDOS		TEXT PROCESSOR	35.00
SCICALC	885-8027	HDOS		UTILITY	20.00
SMALL BUSINESS PACKAGE	885-1071-37	HDOS		BUSINESS	75.00
SMALL-C COMPILER	885-1134	HDOS		LANGUAGE	30.00
SOFT SECTOR SUPPORT PACKAGE	885-1127-37	HDOS		UTILITY	20.00
STUDENT'S STATISTICS PACKAGE	885-8021	HDOS		EDUCATION	20.00
SUBMIT (Z80 ONLY)	885-8006	HDOS		UTILITY	20.00
TERM & HTOC	885-1207-37	CPM		COMMUNICATION & UTILITY	20.00
TINY BASIC COMPILER	885-1132-37	HDOS		LANGUAGE	25.00
TINY PASCAL	885-1086-37	HDOS		LANGUAGE	20.00
UDUMP	885-8004	HDOS		UTILITY	35.00
UTILITIES	885-1212-37	CPM		UTILITY	20.00
UTILITIES BY PS	885-1126	HDOS		UTILITY	20.00
VARIETY PACKAGE	885-1135-37	HDOS		UTILITY & GAMES	20.00
WHEW UTILITIES	885-1120-37	HDOS		UTILITY	20.00
XMET ROBOT X-ASSEMBLER	885-1229-37	CPM		UTILITY	20.00
Z80 ASSEMBLER	885-1078-37	HDOS		UTILITY	25.00
Z80 DEBUGGING TOOL (ALDT)	885-1116	HDOS		UTILITY	20.00

H8 - H/Z-89/90 - H/Z-100 (Not PC)

ADVENTURE	885-1222-37	CPM		GAME	10.00
BASIC-E	885-1215-37	CPM		LANGUAGE	20.00
CASSINO GAMES	885-1227-37	CPM		GAME	20.00
CHEAPCALC	885-1233-37	CPM		SPREADSHEET	20.00
CHECKOFF	885-8011-37	CPM		CHECKBOOK SOFTWARE	25.00
COPYDOS	885-1235-37	CPM		UTILITY	20.00
DISK DUMP & EDIT UTILITY	885-1225-37	CPM		UTILITY	30.00
DUNGEONS & DRAGONS	885-1209-37	CPM		GAMES	20.00
FAST ACTION GAMES	885-1228-37	CPM		GAME	20.00
FUN DISK I	885-1236-37	CPM		GAMES	20.00
FUN DISK II	885-1248-37	CPM		GAMES	35.00
GAMES DISK	885-1206-37	CPM		GAMES	20.00
GRADE	885-8036-37	CPM		GRADE BOOK	20.00
HRUN	885-1223-37	CPM		HDOS EMULATOR	40.00
HUG FILE MANAGER & UTILITIES	885-1246-37	CPM		UTILITY	20.00
HUG SOFTWARE CATALOG UPDATE #1	885-4501	VARIOUS		PRODUCTS 1983 THRU 1985	9.75
KEYMAP CPM-80	885-1230-37	CPM		UTILITY	20.00
MBASIC PAYROLL	885-1218-37	CPM		BUSINESS	60.00
MICRONET CONNECTION	885-1224-37	CPM		COMMUNICATION	16.00
NAVPROGSEVEN	885-1219-37	CPM		FLIGHT UTILITY	20.00
REMARK VOL 3 ISSUES 24-35	885-4003	N/A		1982	20.00
REMARK VOL 4 ISSUES 36-47	885-4004	N/A		1983	20.00
REMARK VOL 5 ISSUES 48-59	885-4005	N/A		1984	25.00
REMARK VOL 6 ISSUES 60-71	885-4006	N/A		1985	25.00
REMARK VOL 7 ISSUES 72-83	885-4007	N/A		1986	25.00
SEA BATTLE	885-1211-37	CPM		GAME	20.00
UTILITIES BY PS	885-1226-37	CPM		UTILITY	20.00
UTILITIES	885-1237-37	CPM		UTILITY	20.00

Price List

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
X-REFERENCE UTILITIES FOR MBASIC	885-1231-[37]	CPM	UTILITY	20.00
ZTERM	885-3003-[37]	CPM	COMMUNICATION	20.00

H/Z-100 (Not PC) Only

ACCOUNTING SYSTEM	885-8048-37	MSDOS	BUSINESS	20.00
CALC	885-8043-37	MSDOS	UTILITY	20.00
CARDCAT	885-3021-37	MSDOS	BUSINESS	20.00
CHEAPCALC	885-3006-37	MSDOS	SPREADSHEET	20.00
CHECKBOOK MANAGER	885-3013-37	MSDOS	BUSINESS	20.00
CP/EMULATOR	885-3007-37	MSDOS	CPM EMULATOR	20.00
DBZ	885-8034-37	MSDOS	DBMS	25.00
ETCHDUMP	885-3005-37	MSDOS	UTILITY	20.00
EZPLOT II	885-3049-37	MSDOS	PRINTER PLOTTING UTILITY	25.00
GAMES CONTEST PACKAGE	885-3017-37	MSDOS	GAMES	25.00
GAMES PACKAGE II	885-3044-37	MSDOS	GAMES	25.00
GRAPHICS	885-3031-37	MSDOS	ENTERTAINMENT	20.00
HELPSCREEN	885-3039-37	MSDOS	UTILITY	20.00
HUG BACKGROUND PRINT SPOOLER	885-1247-37	CPM	UTILITY	20.00
KEYMAC	885-3046-37	MSDOS	UTILITY	20.00
KEYMAP	885-3010-37	MSDOS	UTILITY	20.00
KEYMAP CPM-85	885-1245-37	CPM	UTILITY	20.00
MAPLE	885-8023-37	CPM	COMMUNICATION	35.00
MATHFLASH	885-8030-37	MSDOS	EDUCATION	20.00
ORBITS	885-8041-37	MSDOS	EDUCATION	25.00
POKER PARTY	885-8042-37	MSDOS	ENTERTAINMENT	20.00
SCICALC	885-8028-37	MSDOS	UTILITY	20.00
SKYVIEWS	885-3015-37	MSDOS	ASTRONOMY UTILITY	20.00
SMALL-C COMPILER	885-3026-37	MSDOS	LANGUAGE	30.00
SPELL5	885-3035-37	MSDOS	SPELLING CHECKER	20.00
SPREADSHEET CONTEST PACKAGE	885-3018-37	MSDOS	VARIOUS SPREADSHEETS	25.00
TREE-ID	885-3036-37	MSDOS	TREE IDENTIFIER	20.00
USEFUL PROGRAMS I	885-3022-37	MSDOS	UTILITIES	30.00
UTILITIES	885-3008-37	MSDOS	UTILITY	20.00
ZBASIC DUNGEONS & DRAGONS	885-3009-37	MSDOS	GAME	20.00
ZBASIC GRAPHIC GAMES	885-3004-37	MSDOS	GAMES	20.00
ZBASIC GAMES	885-3011-37	MSDOS	GAMES	20.00
ZPC II	885-3037-37	MSDOS	PC EMULATOR	60.00
ZPC UPGRADE DISK	885-3042-37	MSDOS	UTILITY	20.00

H/Z-100 and PC Compatibles

ADVENTURE	885-3016	MSDOS	GAME	10.00
ASSEMBLY LANGUAGE UTILITIES	885-8046	MSDOS	UTILITY	20.00
BOTH SIDES PRINTER UTILITY	885-3048	MSDOS	UTILITY	20.00
CXREF	885-3051	MSDOS	UTILITY	17.00
DEBUG SUPPORT UTILITIES	885-3038	MSDOS	UTILITY	20.00
DPATH	885-8039	MSDOS	UTILITY	20.00
HADES	885-3040	MSDOS	UTILITY	40.00
HELP	885-8040	MSDOS	CAI	25.00
HEPCAT	885-3045	MSDOS	UTILITY	35.00
HUG BACKGROUND PRINT SPOOLER	885-3029	MSDOS	UTILITY	20.00
HUG EDITOR	885-3012	MSDOS	TEXT PROCESSOR	20.00
HUG MENU SYSTEM	885-3020	MSDOS	UTILITY	20.00
HUG SOFTWARE CATALOG UPDATE #1	885-4501	VARIOUS	PROD 1983 THRU 1985	9.75
HUGMCP	885-3033	MSDOS	COMMUNICATION	40.00
HUGPBBS SOURCE LISTING	885-3028	MSDOS	COMMUNICATION	60.00
HUGPBBS	885-3027	MSDOS	COMMUNICATION	40.00
ICT 8080 TO 8088 TRANSLATOR	885-3024	MSDOS	UTILITY	20.00
MAGBASE	885-3050	VARIOUS	MAGAZINE DATABASE	25.00
MATT	885-8045	MSDOS	MATRIX UTILITY	20.00
MISCELLANEOUS UTILITIES	885-3025	MSDOS	UTILITIES	20.00
PS's PC & Z100 UTILITIES	885-3052	MSDOS	UTILITY	20.00
REMARK VOL 5 ISSUES 48-59	885-4005	N/A	1984	25.00
REMARK VOL 6 ISSUES 60-71	885-4006	N/A	1985	25.00
REMARK VOL 7 ISSUES 72-83	885-4007	N/A	1986	25.00
REMARK VOL 8 ISSUES 84-95	885-4008	N/A	1987	25.00
SCREEN DUMP	885-3043	MSDOS	UTILITY	30.00
UTILITIES II	885-3014	MSDOS	UTILITY	20.00
Z100 WORDSTAR CONNECTION	885-3047	MSDOS	UTILITY	20.00

PC Compatibles

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

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

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

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

Make the no-hassle connection with your modem today! **HUGMCP** doesn't give you long menus to sift through like some modem packages do. With **HUGMCP**, YOU'RE always in control, not the software. Order **HUG P/N 885-3033-37** today, and see if it isn't the easiest-to-use modem software available. 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 and MasterCard phone orders, telephone the Heath Users' Group directly at (616) 982-3463. Have the part number(s), descriptions, and quantity ready for quick processing. By mail, send your order, plus 10% postage and handling (\$1.00 minimum charge, up to a maximum of \$5.00) to: Heath Users' Group, P.O. Box 217, Benton Harbor, MI 49022-0217. VISA and MasterCard require minimum \$10.00 order. No C.O.D.s accepted.

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



The other cats get to sing along!

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

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

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

HEPCAT gets along with everyone . . .

HEPCAT supports more video configurations than any other pop-up, and always

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

HEPCAT works harder . . .

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

these operations: MOD, AND, OR, XOR, SHL, SHR.

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

HEPCAT eats less . . .

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

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

BUGGIN' HUG

Easily Reset Your SmartWatch

Dear HUG:

For years I have used a GC-1000 Most Accurate Clock and when I installed my H-100 (Not PC) I mated my GC-1000 to the H-100. Recently, I decided to simplify things and add an FBE Smartwatch to my H-100. I have not yet abandoned the GC-1000, but instead have been checking the accuracy of the Smartwatch. It is supposed to be accurate to within one minute per month, and it is.

This means that the SmartWatch must be reset every so often. Resetting is not too complicated. But when I discovered how simple it is to correct for the Daylight Savings change I decided that it would be nice to have a simple way to adjust the watch's time. I can now add ONE Minute or Subtract ONE Minute by the Command "DSCLOCK +", or "DSCLOCK -".

So I thought that HUG readers may be interested in knowing how I did it. It is easy to do because the Software Disk that is provided with the FBE SmartWatch included the source assembly file. Using an editor (I use BSE), revise DSCLOCK.ASM as follows:

Find the label "NOTSPC:". Erase the label leaving the rest of the line intact (CMP AL,"D" ;DST Command?). Immediately above add four lines as follows:

```
NOTSPC:  CMP AL, '-'
        JE ADJSUB
        CMP AL, '+'
        JE ADJADD
```

The next line will be the above mentioned line: CMP AL,'D'. Drop down to the DOLINE routine. Immediately above it add the following two routines:

```
ADJSUB:  CALL RDCLK      ; Get Current Data RFH 4/3/89
        MOV AL,MINS
        CMP AL,00       ; Is Packed Decimal
        JE ADJSU1
        SUB AL,01
        DAS
        MOV MINS,AL
        JMP WRCLK       ; Complete Adjustment
ADJSU1:  MOV AL,HOURS
        CMP AL,00
        JNE ADJSU2
        RET             ; Don't touch it!
ADJSU2:  MOV MINS,059H   ; Packed Decimal
        SUB AL,01
        DAS
        MOV HOURS,AL
        JMP WRCLK
```

```
ADJADD:  CALL RDCLK      ; Get Data RFH 4/3/89
        MOV AL,MINS
        CMP AL,059H     ; Is Packed Decimal
```

Name Change of the Phoenix HUG

Dear Jim:

As President of the Phoenix Heath/Zenith Users' Group (PHUG), it is with remorse that I inform you of the termination of our group (PHUG). With attitude changes at our local Heath/Zenith Electronics Center, this change in our status has been pending for sometime. The majority of our group still uses Heath/Zenith computers and software. Although the group is moving to a new meeting location and has changed the name to Phoenix Computer Group, PCG, we will render assistance to anyone from the Heath/Zenith community that needs our help. Our BBS (602) 254-2185 still supports MS-DOS, ZDOS, CP/M and HDOS.

In closing, let me state that it has been fun, as well as educational over the past 8 years dealing with HUG. We wish the Best of Luck to the remaining HUG members across the country.

Sincerely,
Ron Eggemeyer, President
Phoenix Computer Group

What's the Purpose of HUG

Dear HUG:

I will probably renew my membership and REMark subscription even if you do not incorporate any of my suggestions, but I thought that since I was not included in your recent member survey, I would submit my two cents' worth:

What is the purpose of HUG and REMark? I think this organization should be aimed solely at Heath/Zenith hardware and Zenith MS-DOS compatibility with commercial hardware/software, ZDS tips, ZDS product improvements, problems, solutions and limited offerings of HUG software. Your new feature Bug Zapping is a great idea and a start in the

right direction.

I subscribe to, and read, numerous general and special purpose PC and software specific trade publications. I do not need another magazine that reviews software or general hardware products. Specifically, in the March 89 edition of REMark, I did not need the articles on XyWrite, WordStar 2000, WordPerfect 5.0, and Printers. Those articles could have been published in any trade magazine and had little or no bearing on Heath/Zenith products.

The Enable article could be helpful only because no other widely-circulated publication writes about Enable and because so many DoD agencies adopted it since Zenith included it on the Desktop II contract.

Instead of including general purpose software articles, I think REMark should devote itself to detailing any problems and solutions using commercial hardware and software with Zenith PC desktop and laptop computers and monitors and should be our collective voice in resolving these problems with either Zenith or the third party.

For instance, WordPerfect 5.0 has had several problems with a number of non-IBM ROM BIO chips (including Zenith) and their cursor command feature. These problems finally appear to have been solved by the 1-3-89 release of WP. Further, WordPerfect will not work with the Zenith 449 enhanced EGA video card unless the auto-emulation switch 5 is turned off. Other software, including Charts, has the same problem, but Enable 2.15 appears to work fine with the auto-emulation switch on.

Banyan's Vines 286 network software does not work with the Zenith MS-DOS 3.21 VDisk. You must use VDisk.sys from DOS 3.2 or earlier. I am sure that there have been problems installing certain third party disk drives and memory cards. These are real world problems that HUG could help us with.

```
JE ADJAD1
ADD AL,01
DAA
MOV MINS,AL
JMP WRCLK      ; Complete Adjustment
ADJAD1:  MOV AL,HOURS
        CMP AL,023H
        JNE ADJAD2
        RET     ; Don't touch it!
ADJAD2:  MOV MINS,00
        ADD AL,01
        DAA
        MOV HOURS,AL
        JMP WRCLK
```

Now reassemble, relink, and convert from EXE to COM.

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

You should review every new release of Zenith ROM BIOS chips, MS-DOS, BASIC, and Diagnostics software and advise us of problems, recommendations, and ordering information (both by mail or by phone for those of us that do not live near Heath stores).

A brief review of new Zenith computer products would be helpful, but generally Zenith hardware and software is simply over priced. Therefore, I think it is entirely appropriate for your authors to make recommendations regarding third party peripherals that work with Zenith hardware and compare them to the official Zenith products.

You should be our voice to complain about Zenith versions of commercial software, such as Windows. Zenith's updates are always very late and over priced. Further, the Zenith version sometimes leaves out features, such as PC Write from Windows, and the software manufacturer will not allow us to upgrade directly from the Zenith version.

Sincerely,
Craig Carver
1602 Tulagi Street
Barstow, CA 92311-5944

Xplore Your Batch Files . . . And Xplode Some Myths

Dear HUG:

This article contains a line in the listing for ASSEM.BAT that can cause the user of the batch file to get strange error messages if used with LINK.EXE as distributed with MS-DOS 3.3+, QuickBasic 4.0, and perhaps other late versions of LINK. It is, of course, the line, LINK SHOW.OBJ, SHOW.EXE, NUL, NUL.

This was a take-off from a line in a batch file that I have used for years with versions of LINK from Z-DOS and on, where it had the form, LINK %1.OBJ, %1.EXE,NUL,NUL.

Now, these newer versions of LINK choke up over that line. The problem is with the use of NUL as the last argument, to show no library file. If the line is changed to:

```
LINK %1.OBJ,%1.EXE,NUL; or to,  
LINK %1.OBJ,%1.EXE; or to,  
LINK %1.OBJ;  
no error messages are generated.
```

When I discovered that the MS-DOS 3.3+ LINK has this unfortunate difference from earlier versions, I contacted the local Heath/Zenith store, since I had acquired the DOS with a SupersPort laptop purchased there. I didn't really expect any remedy other than perhaps advice as to other ways of avoiding the problem, but thought that I could feed back some information that might save other users the trouble of digging out the cause. The response was very disconcerting, a total lack

of interest, and an assertion that the reason for the lack of interest was that "programmers are not that large a part of our customer base."

This might be an item for your Bug Zapping feature, or if you can put it in your letters to the editor, you might save your readers a lot of trouble. The error messages that result are of the "red herring" type that cause the waste of a lot of time before you start looking in the right place. The DOS 3.3+ LINK gives the message "Invalid object module" and QuickBasic LINK gives "fatal error L1102: unexpected end of file."

Sincerely,
Robert G. Brasfield
10344 Dibble Avenue, N.W.
Seattle, WA 98177

Thanking the SYSOP

Dear Jim:

On April 7, 1989, I got to thinking that I should logon to HUGBBS and check out the Bargain Centre and check out what good stuff was there since my last time. I also got to thinking that lots of the times I forget to thank the SYSOP for the work he is doing. I thought that to save time on the ole phone line and to not keep someone else waiting to use the board, I would just compose a message to the SYSOP offline and send it to HUGBBS one line at a time at the prompt asking to leave a message to the SYSOP. I thought this would work since I had used it on several boards before. Of course, I tried to keep it short and kinda to the point. Well, it seems that HUGBBS does not like messages to the SYSOP of more than 5 lines. Just as I sent line #5 HUGBBS abruptly sent me a "Goodbye, And Thank You For Calling . ." message and turned off the carrier. What gives? Generally, if the message is too long, the system will state so or send back a "buffer full" message to the terminal.

I still wanted to get the message to you; hence, this letter.

Jim, you are doing a great job with the HUG bulletin board . . . I have noted how fast the board reacts to being linked with a remote terminal. I am not sure just how HUGPBBS does this. Generally, the HOST will have to be sent several CR's, but not this one. I have not had any problems linking up and staying on line, i.e., losing carrier extraneous characters and the like.

I hope the recent Bargain Centre sale has been a success . . . I ordered one of the Z-159 CPUs . . . haven't received it or heard from HUG . . . I'm just impatient, I guess.

Others have claimed that their BBS is the fastest, but even though this little mishap occurred, I still think that not

only is HUGBBS a fast system, but is quite accurate. And the SYSOP is doing a "slam-bang" good job.

Sincerely,
Allie C. Lingo
P.O. Box 118
Dierks, AR 71833-0118

Ed: Thanks for the kind words Allie. The message to the SYSOP on HUGPBBS does only allow 5 lines of text.

Bug Zapping Item

Dear HUG:

It was with a great deal of interest I read the February 1989 Bug Zapping item on H-150 keyboard problems. However, when I went to install the 270 pF capacitor as suggested, I discovered my CPU board had different labels from those mentioned in the article. My CPU board has a type-written label of 181-5326 and a permanent label of 85-2889-1. Does anybody have a fix for this board variation?

Regards,
Tony Thurston
Rivendell Farm
RR #2
Kinburn, Ontario
CANADA K0A 2H0



**EXPLORE
NEW WORLDS
WITH
HUG
GAME
SOFTWARE**

All Speed and No Fluff

Pat Swayne
HUG Software Engineer

A Review of the AOX Z-Master 386 Upgrade (Also: The ZX-386 Revisited)

Zenith Data Systems must have sold quite a few Z-248's, considering the interest there is in providing upgrades for it. As of this writing, there are now four available. In the January issue of REMark, I reviewed the ZX-386 upgrade from American Micronics, Inc., and I also made considerable mention of the HUG-386 upgrade in that article. In addition to those upgrades, there is the First Capitol Computer upgrade, which is really a trade-up (since you send in your old computer and get a new one back); and there is the AOX Z-Master 386 upgrade, which is the subject of this review. Because the Z-Master 386 and the ZX-386 are the most similar of the four upgrades available, this review will include a considerable amount of head-to-head comparison between the two.

A Description of the Z-Master 386

The Z-Master 386 is very similar to the ZX-386 in that it is a drop-in replacement for the Z-248 CPU card. Unlike the ZX-386 system, which is a two board system (a mother and daughter board connected together) that replaces both CPU and I/O cards, the Z-Master 386 is a single card system that retains all other existing Z-248 cards. You can even use your Z-248 memory cards, which will not work with the ZX-386 (though standard AT memory cards will).

Like the ZX-386, the Z-Master 386 provides high speed 32-bit memory op-

tions ranging in size from 1 megabyte to 8 megabytes. Unlike the ZX-386, which uses standard memory chips on its daughter board, the Z-Master 386 uses the newer SIMM memory modules. You can put 1 megabyte or 4 megabytes of memory on a standard Z-Master 386 depending on whether you use 256k or 1 megabyte SIMMs. The SIMMs are plugged into sockets that hold them at a sharp angle, so that the total width of the board, plus memory modules, is not greater than the width allowed for a single board. The printed circuit board is actually laid out to accept 8 memory modules, but there is only room to mount four of them when they are socketed. Therefore, if you buy a standard Z-Master 386 board and later want to upgrade it to 8 megabytes, you must send it to the factory. There they will solder special SIMMs (with solder tail connectors rather than socket connectors) directly to the board.

The Z-Master 386 provides a feature that is not provided by any of the other upgrades in their standard configurations — a high speed cache. A cache is a special block of high speed memory that is used to hold computer instructions that are executed repeatedly, as in a loop. For a more detailed explanation of a cache memory system, see page 38 of the July 1988 issue of REMark. The Z-Master is a remarkable engineering achievement, when you consider that it is the equivalent of up to 4 Z-386 cards (a CPU card, a

cache memory card, and 2 4-megabyte memory boards), all packed on to a single card.

The Z-Master 386 is available in 3 speed versions — 16, 20 and 25 MHz. The 25-MHz version is the fastest Z-248 upgrade available.

Installing the Z-Master 386

The Z-Master 386 is supplied with a thin spiral bound manual. The manual fairly well written, and I did not find any glaring mistakes in it. However, it does not mention that you must return a board equipped with 1 or 4 megabytes of memory to the factory if you want to upgrade to 8 megabytes. I found that information out by calling AOX. They always answered their phone promptly when I called, and returned calls when the person having the answer to a question was not available.

The manual also did not mention that before you remove your Z-248 CPU card, you should note the Setup information about your hard disk (heads, cylinders, etc.). As with the ZX-386 board, the number for your drive type in the Z-Master 386 Setup may be different from the number in the Zenith Setup.

Installation is simply a matter of removing the old CPU card and installing the Z-Master 386 card. There is a DIP switch module containing 8 switches that must be set properly before the computer is powered up. Four of the switches are used to set up the memory configuration.

There are switch settings listed for 1 and 4 megabytes of 32-bit memory, and various amounts of Zenith memory, but there are no settings listed for 8 megabytes of 32-bit memory, or for standard AT-type memory boards. In fact, there is no mention of whether standard AT memory boards will work or not. An AOX representative told me that they do work, however, and are addressed above any Z-Master and Zenith memory installed. In the case of the 8 megabyte board, special instructions would probably come with it as to the switch settings (if they were different from the 4 megabyte settings). The whole business of having memory switch settings appears to be mainly to support Zenith memory cards (except for 1 switch, to switch between 256k or 1 megabyte SIMMs). The AMI ZX-386 board, which does not support Zenith memory boards, does not have any memory switch settings.

If you use Zenith memory boards, you must install the backfill jumper on the memory board, J405, if you want 640k of base memory. Even though the base memory is filled entirely with 32-bit memory from the Z-Master card, the jumper must still be installed. When you set the switches on the Z-Master 386 card to indicate the amount of Zenith memory installed, they should be set to the amount of non-backfill memory, even though some or all of the backfill memory may be used somewhere in the memory map by the Z-Master 386. If you fail to set the switches correctly, you will find yourself cheated out of some of your memory. I found out these things by experimenting, as the Z-Master 386 manual is unclear on exactly how to set up the switches. An AOX representative told me that a new manual is on the way with clearer switch instructions. He also told me that AOX has developed a program that can be run on your H/Z-248 before you install the Z-Master 386, which will examine your installed memory and indicate how you should set the Z-Master switches.

Two of the remaining switches on the Z-Master 386 board are used to control how the cache works. You can cache your RAM memory only, or any combination of RAM memory, the BIOS ROM, and the video ROM. Usually, you would want to cache all three of these, but if you have a Z-449 card (which has turned out to be the Edsylv of Zenith video cards), you must turn video caching off. When I first got the Z-Master 386, it would not work at all with the newer Z-549 card (an analog-only version of the HVB-550 card found in the latest Heathkit catalog). AOX recalled the board for modification, and when it was returned, it worked fine with the Z-549. The video cache switch was off when the board was returned, but it seems to work fine with it on.

The last two switches are to set

whether your EGA (or VGA) card has a Zenith ROM or not, and whether you will be running Xenix or not. The switch disables the cache, which does not work during Xenix boot-up. A device driver supplied by AOX can turn on the cache after boot-up. They have device drivers for both Xenix and MS-DOS, in case you need to run both operating systems (which would require that the cache be turned off at the switch).

You can install the Z-Master 386 into a Z-286 (not Z-286 LP) and no modifications to the cabinet are required (modifications were required with the ZX-386). All you have to do is to replace the mounting bracket with the taller one from your CPU card.

When you first power up the Z-Master 386, a count of both the base memory installed and the extended memory will be displayed, and then you will be prompted to run Setup if the memory configuration is wrong. Since the Setup information is stored in the I/O card on a standard Z-248 system, your old Setup information will still be intact, so the only thing you will probably have to change is the memory size figures. But you should be sure to check your hard disk type and make sure that the parameters for the type number shown are the proper ones for your drive.

The BIOS used in the Z-Master 386 is a Phoenix BIOS, probably the most popular BIOS used in "clone" computers. It is a very plain-vanilla BIOS, with no extra goodies, such as test utilities or a debugger, included. The operation of Setup is a little more like Zenith's than AMI's setup in that you use the arrow keys to move a marker to the various parameters, and then other keys are used to change the values. The memory sizes, however, are typed in as numbers. As with the ZX-386 board, there is no way to change the way the computer boots up. It always boots up in the IBM style, by trying floppy drive A: first, and then booting the hard disk.

One interesting feature of the Z-Master 386 is that it gives you all of your first megabyte of memory. If you only have 1 megabyte of 32-bit memory in the system, the startup memory count will indicate that you have 640k of base memory and 384k of extended memory. In a Z-386, you can use 256k of your first megabyte beyond the 640k of base memory as EMS memory, but you cannot use any of it if you elect not to have EMS memory. The Z-386 uses 128k of the first megabyte itself, for "slushware" (the system ROMs are copied to RAM memory for faster operation). The ZX-386 does not let you use any of the first megabyte beyond the 640k base memory. Newer versions use some of it for "ROM shadowing", which is the same thing as slushware.

Performance Tests

The Z-Master 386 has been a solid

performer since installation, and the only problem encountered was operation with the Z-549 card, which has been corrected. Every program I have tried has run perfectly, with the exception of speed dependent games.

Those of you who have read my previous articles on upgrade modifications know that I use a battery of different tests, in order to present as fair a comparison between various systems as possible. My tests include three speed computation programs, two operational tests, and a video speed test. As I mentioned previously, the AMI ZX-386 upgrade has been modified to support ROM shadowing, so I have redone the ZX-386 tests, and will include the results here. As usual, I have also included the test results for an H-248 and an H-386. (Note: the H-248 is the Heathkit version of the Z-248, and the H-386 is the Heathkit version of the Z-386.)

The H-248 used in these tests runs at 8 MHz with no memory wait states. The H-386 runs at 16 MHz with variable wait states (paged memory system). The ZX-386 and the Z-Master 386 are both the 20 MHz versions, and they probably use a variable wait state memory system. Little technical detail is provided with either board.

The speed computation programs I use are the Norton SI (System Information) program (version 3.00), the PC Tools Info command (version 3.24), and my own SPEED program (version 1.0), from HUG disk no. 885-3052. The results returned by each of these programs are supposed to be a comparison with the original IBM PC. Here are the SI test results.

Norton SI Ratings (IBM PC = 1.0)

System	Rating
H248	9.2
H386	18.3
H-386 with Z-525 cache	18.7
ZX-386	23.0
ZX-386 with shadowing	23.0
Z-Master 386	30.5

(The Z-Master 386 actually returned 29.7 part of the time and 31.6 part of the time during several tests.) As I have stated in previous articles, the Norton SI test gives a rather inflated rating for processors other than an 8088. Below are the PC Tools Info test results.

PC Tools Info Ratings (IBM PC=100)

System	Rating
H-248	460
H-386	735
H-386 with Z-525 cache	780
ZX-386	860
ZX-386 with shadowing	945
Z-Master 386	1030

Here are the SPEED ratings. The SPEED program performs a prime number calculation, and is a good indicator of the "raw" computational power of a computer.

SPEED Ratings (IBM PC=100)

System	Rating
H-248	528
H-386	1056
H-386 with Z-525 cache	1192
ZX-386	1409
ZX-386 with shadowing	1409
Z-Master 386	1720

(The Z-Master 386 actually returned 1660 part of the time and 1772 part of the time during several tests.) Notice that the Z-Master 386 is faster in all of these tests, even though it has the same clock speed as the ZX-386. Notice also, that ROM shadowing only made a difference in the Info test. This test apparently makes some kind of BIOS calls, because only BIOS operations are affected by shadowing.

While the above tests are good for making general comparisons between systems, they do not take peripheral devices into account. A computer must access storage devices and its video output while it is doing "real" work, so I performed two tests that are actual computer jobs. The first test is to assemble the HUG program HADES (HUG Absolute Disk Editing System) using MASM version 4.0, and here are the results.

Source Code Assembly	Time (sec)
H-248	27.1
H-386	16.0
H-386 with Z-525 cache	14.1
ZX-386	14.2
ZX-386 with shadowing	13.1
Z-Master 386	11.8

This test reveals that while ROM shadowing provides some increase in performance, a high speed cache helps the most. This test involves internal calculations (by the assembler) and disk access. The next test was to have AutoCAD version 2.62 load and draw a sample drawing. Here are the results.

AutoCAD Drawing	Time (sec)
H-248	22.8
H-248 with 80287	13.6
H-386	14.9
H-386 with Z-525 cache	11.6
ZX-386 and Z-449	12.4
ZX-386 w/shadowing and Z549	11.4
Z-Master 386 and Z-449	11.6
Z-Master 386 and Z-549	11.4

All of these tests were done with a Z-449 card installed, except those that indicate a Z-549 card. The ROM on the Z-449 card cannot be cached on the Z-Master 386, and it cannot be shadowed on the ZX-386. The fact that the ZX-386 was able to keep up with the Z-Master 386 in this test gives us a clue of the affect that ROM shadowing has on video response. Notice also that the slower 16-

MHz Z-386, when equipped with a Z-525 cache, was almost able to keep up. This shows the affect of a combination of caching and ROM shadowing (slushware).

To find out the affect that ROM shadowing and caching have on video response, I developed a little program called VIDTEST. The assembly source code for VIDTEST can be found on page 55 of the January issue of REMark. Vidtest performs two tests that were recommended in PC Magazine. In the first test, the screen is cleared and then filled with 24 60-character lines. This process is repeated 10 times. Here are the results of this test.

Video Test One	Time (sec)
H-248 (Z-449)	11.0
H-386 (Z-449)	2.7
H-386 with Z-525 cache	2.3
H-386 (Z-549) with cache	1.5
ZX-386 (Z-449)	10.1
ZX-386 (Z-549)	9.0
ZX-386 (Z-549) w/shadowing	1.5
Z-Master 386 (Z-449)	12.0
Z-Master 386 (Z-549)	1.7

Notice the remarkable difference in the times for the Z-Master 386 with the Z-449 card and the Z-549 card. As I stated previously, the ROM on the Z-449 cannot be cached on the Z-Master 386. You can see by comparing the Z-449 card times on the Z-Master 386 and the H-386 that it is the caching, not the difference in the video cards themselves, that produces most of the improvement seen in the second Z-Master 386 result.

The best times for the first test were produced by the ZX-386 with ROM shadowing enabled and a Z-549 card, and the H-386, which uses both ROM shadowing and caching. While caching seems to have a greater affect in improving overall performance, ROM shadowing seems to help the best in improving video response. This is further demonstrated by the second video test. In this test, 240 60-character lines are written to the screen, causing the screen to scroll as lines are added. Here are the results of this test.

Video Test Two	Time (sec)
H-248 (Z-449)	14.5
H-386 (Z-449)	4.4
H-386 with Z-525 cache	4.0
H-386 (Z-549) with cache	2.7
ZX-386 (Z-449)	12.1
ZX-386 (Z-549)	10.2
ZX-386 (Z-549) with shadowing	2.7
Z-Master 386 (Z-449)	14.9
Z-Master 386 (Z-549)	4.6

As you can see, this test was won easily by the computers having ROM shadowing — the H-386 and the ZX-386.

Those of you with Z-449 cards may

be shaking your heads and thinking that you may have to get another video card if you get one of these upgrades and you want snappy video performance. As I mentioned in my article on the ZX-386, you can make the Z-449 run the above two tests as fast as it does on an H-386 if you use video speed-up software such as VSCREEN. Unfortunately, VSCREEN is no longer included with the Mace Utilities, as I stated before. Also, the version of VSCREEN that I had turned out to have bugs in it. Later versions of VSCREEN may not have the bugs, and the program can still be obtained from its original supplier, Golden Bow Systems. There are also other screen speed-up programs available, including one I wrote. Mine can be found in a file called AMI-AOX.ZIP, which is on the HUG bulletin board (615-982-3956).

The Serial Printer Problem

As I mentioned in my review of the ZX-386, the LPT to COM port mapping feature of Zenith's CONFIGUR does not work with a non-Zenith BIOS. So if you have a serial printer, it will not work on a Z-Master 386 system without some kind of additional software support. I have written a little program called MAPLPT, which is included in the previously mentioned AMI-AOX.ZIP file. That file is a "compressed" file, which means that several files have been combined into one for easier downloading. You will need PKUNZIP (also available on the HUG bulletin board) to extract the individual files.

The ZX-386 Revisited

Since my review of the ZX-386, it has been modified somewhat to improve its performance, and I feel that I should describe those modifications here. As I mentioned earlier, it now supports "ROM shadowing", which means that the code in the ROMs is copied to faster RAM memory at startup. Both the system ROM and the video ROM can be shadowed. To make it possible for you to enable and disable shadowing, an extended setup feature has been added to the system ROM. AMI uses a chip set from the Chips and Technologies company, and their new extended setup feature allows you to control the way those chips work. In addition to being able to control ROM shadowing, you can also enable or disable memory interleaving, and there is a special "high level" setup that allows you to control the actual registers within the C and T chips. Unfortunately, as of this writing they have not produced a new manual that explains any of this. So I had to leave the high level setup feature alone. I suspect that I could tweak a bit better performance out of the ZX-386 if I knew what I was doing.

If you have a ZX-386 with the new extended setup feature, I would recommend that you enable shadowing for the

system ROM and the video ROM (unless you have a Z-449 card), and I would recommend that you enable memory interleaving. My tests were done with the board set up that way.

The price that you must pay for the new extended setup feature is that it takes the place of the test features that were in the old system ROM. All of the built-in diagnostic tests that I described in my January article, including the hard disk interleave test, are now gone.

Pluses and Minuses

The Z-Master 386 has been a solid performer, and well worth your consideration if you are thinking of upgrading your '248. On the plus side, here are some factors to consider:

- Easy installation. The Z-Master is the easiest of all the upgrades to install. The switch settings were a bit confusing, but a new manual is promised that should clear them up.
- Fastest operation. With a 25 MHz version available, the Z-Master 386 is the fastest H/Z-284 upgrade available. The built-in high speed cache makes even the 20 MHz version faster than the other upgrades tested.
- Uses H/Z-248 memory. The Z-Master 386 is the only upgrade that lets you keep your Z-415 or Z-445 memory boards.
- Works in a Z-286 (not Z-286 LP). The Z-Master 386 can be installed in a Z-286 with replacement of the mounting bracket, the only modification required.

On the negative side, here are some things to consider.

- No Zenith BIOS. The Phoenix BIOS may be famous throughout the non-Zenith clone world, but it is about as "plain Jane" as a BIOS can be. A Zenith BIOS provides the ability to boot from

alternate floppy and hard drives, a built-in debugger and test utilities, and serial printer support. The latter two features can be provided by external software, but I doubt that it is possible to add the alternate drive boot feature.

- No ROM shadowing (slushware). The Z-Master 386 would be even faster if AOX like AMI, could add ROM shadowing to their system.
- The only full performance 32-bit memory available to the system is the memory on the Z-Master 386 board, and the most memory you can install yourself is 4 megabytes. To upgrade to 8 megabytes from 1 or 4 megabytes, you must send your board to the factory.

Recommendations

For most users, I would recommend the Z-Master 386 upgrade as the way to go to upgrade an H/Z-248. It is the fastest of the upgrades we have tested, and it is the easiest to install. And AOX is offering a 20% discount to HUG members, which should make the Z-Master especially attractive.

My own personal preference is the AMI ZX-386, because as a tinker and experimenter, I appreciate the ability to change speeds and wait states on the fly

for the Z-Master 386, but I was not supplied that utility for the Z-Master 386, and am unable to report on it or compare it with the speed change capabilities of the ZX-386 or Z-386.

If you want a real Zenith machine when you get through, the HUG-386 is the only way to go. The fact that a 16-MHz H-386 can keep up with the 20 MHz versions of the other upgrades in some of the tests is a testament to the superiority of Heath/Zenith technology. And with the HUG member discount, the price is competitive. But Zenith memory boards are expensive, so keep that in mind as you compare systems.

Where To Get It

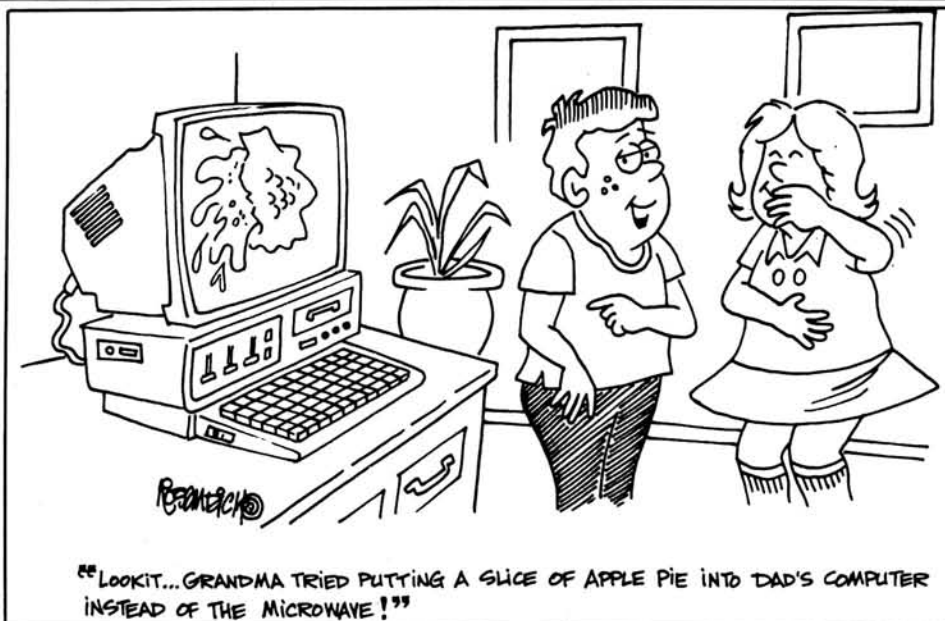
The Z-Master 386 is a product of:
AOX Incorporated
486 Totten Pond Road
Waltham, MA 02154
(617) 890-4402

To get the 20% HUG member discount, call AOX and tell them you are a HUG member. They will provide the name of a participating distributor who will sell the Z-Master 386 at the discount. Here are the prices of some selected models. All units listed have no (0k) memory installed.

Item	Standard Price	Discount
16 MHz Z-Master 386	\$1495	\$1196
20 MHz Z-Master 386	\$1750	\$1400
25 MHz Z-Master 386	\$2195	\$1756

(you can do it with special key combinations), and the extended configuration option. The ZX-386 is almost as easy to install as the Z-Master 386, unless you have to separate the daughter board from the mother board (to install an 80387 or change the video jumper), which can be difficult for some people to do safely. (An AOX representative informed me that they have a software speed change utility

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



Programmer's Learning Corner

Don Keller
1330 Eden Valley Road
Port Angeles, WA 98362

The Bouncing Highlight

One programming trick, above all others, makes a good impression on casual computer users and that's a menu system that displays choices with ONE of them highlighted (by reverse video for instance). Pressing the ENTER or RETURN key selects the choice that's highlighted. Pressing the space bar moves the highlight to the next entry and pressing the backspace key moves it to the previous entry. It's a technique that lends itself nicely to hierarchical menus; selecting one entry can reveal a new set of choices and one of those can bring up another new set and so on. At the same time, the user is never confronted with a bewilderingly large menu. If a selection can also be made by pressing the key that corresponds to the initial character of any visible menu entry it's even better. After a few sessions with a program designed to work that way, inexpert users begin to acquire feelings of sophistication and control. And friends, when you want somebody to come back for ANOTHER program, that's the effect you'd better get; which is why big software shops use that trick. Custom programmers often lack the library support to do the same thing.

The listings presented here describe a C language library to make a highlight menu system reasonably easy to program. You shouldn't need to copy more than about half of it for any single computer system. Multiple versions of support functions are provided to work with Eco_C in MS-DOS, and C/80 in either CP/M or HDOS. Manipulating a display screen involves a certain amount of unavoidable operating system dependence. Other compilers should work too, with a little judicious tinkering in response to the compiler's complaints about the way the source code is written.

```
/*      menuf.h      Listing 1
*/

#define FORE      '\040' /* space bar, moves highlight to next selection */
#define BACK      '\010' /* backspace, moves highlight to previous selection */

struct menus {          /* menu structure template */
    int y, x;
    char *s;
};

/*#define stderr 0 /* !!!!! define this ONLY for C/80 in CP/M or HDOS !!!!! */
```

Listing 1 is a header file that contains a few definitions needed throughout the library. It's commented well enough to let you use it. The values defined for 'FORE' and 'BACK' can be changed if you'd rather use some other pair of key values to move the highlight back and forth through the menu. That structure template defines the word 'menus' as a structure tag and establishes the kind of structure 'menus' refers to. The last line in the header file is for eight-bit C/80. If that's what you're using, the line should be copied without the leading '/*'. Eco_C defines stderr (the standard error output) for you; C/80 does not. Channel 0 is always the terminal in C/80 so defining stderr as 0 approximates a standard error output. It works just fine for most programming, including the utilities described here.

Demenu.c in listing 2 demonstrates a simple menu display.

```
#include <menuf.h>
```

The structure tag in menuf.h is needed in demenu.

```
static struct menus m[] = {
    1,      4,      "",
    22,     12,     "first",
    22,     29,     "second",
    22,     47,     "third",
```

```
22,      64,      "quit"
};
```

This says that m[] is an array of structures of type 'menus'. The header file notified the program that a structure of type 'menus' contains an integer named 'y', an integer named 'x' and a pointer to characters named 's'. The compound statement here goes on to initialize the structure array. That's how you write a menu.

Ignore the first assignment line for a moment (1, 4, "",) and look at the remaining four lines. These are array elements 1, 2, 3 and 4. An integer constant in the leftmost column is a value assigned to the variable named 'y' in the structure. The center column assigns values to 'x' and the right-hand column assigns a character string to the 's' pointer. The assignment in array element 1 goes like this: Starting at line 22 (y = 22) and column 12 (x = 12) the string "first" will be written to the screen as a menu entry (s = "first", actually the address where the string "first" is stored in memory).

Once each entry string is written along with its y and x coordinates, you can discover and insert the values for 'y' and 'x' in element 0 of the array. 'y' is set to the value of the array index for the first

menu entry and 'x' is set to the index of the last entry. Doing it that way lets the menuf() function deal with a variable number of menu entries without adding to the number of arguments passed to menuf(). No string is needed in element 0 of the array but if it's just left out, the order of assignments gets disrupted so the null string is assigned ("").

```
int c;
```

When menuf() returns a value it's stored in 'c'.

```
curoff();
```

You probably don't want your menu messed up with a cursor hanging around in odd places in the display so the curoff function turns it off.

```
tclear();
```

Clears the terminal screen. All of the functions called by demenu.c are provided in the listings included with this article.

```
While (1) {
```

Starts a loop that will execute forever unless something inside the loop causes a breakout.

```
if ((c = menuf(m)) == 4) {
    tclear();
    curon();
    break;
}
```

The value returned by menuf(m) is assigned to 'c'. If it's 4, which is the index number for "quit" up there in the structure array, tclear() wipes everything off the screen, curon() turns the cursor back on and 'break;' gets us out of the infinite loop. There are no more statements following the loop so the program exits to the operating system. The argument 'm' to the menuf function call delivers the information we wrote in the structure array. You can't pass a structure in a function call but you can pass a pointer to a structure. For all practical purposes, an array name is a pointer and that's how we get away with it here.

```
tmove(10, 36);
tputs(" ");
tmove(10, 36);
tputs(m[c].s);
```

The (invisible) cursor is moved up to the middle of the screen, a string of blank spaces is written in case this is not the first write to that area, and the menu selection string is repeated in the new location to show that menuf(m) is actually recognizing specific choices. m[c].s is a pointer to a string associated with the pointer 's' in the structure defined by the tag 'menus' and at element number 'c' in the array 'm[]'. This segment of code gets executed when 'c' does NOT contain a signal to quit. In a serious program you would likely use a switch test to direct program flow according to the selection number returned by menuf(m) but in this demo program let's keep it simple.

```
/*      demenu.c      Listing 2      demonstrate menuf.c and menuf.h
*/

#include <menuf.h>

main()
{
    static struct menus m[] = {
        1,      4,      "",
        22,     12,     "first",
        22,     29,     "second",
        22,     47,     "third",
        22,     64,     "quit"
    };

    int c;

    curoff();
    tclear();
    while (1) {
        if ((c = menuf(m)) == 4) {
            tclear();
            curon();
            break;
        }
        tmove(10, 36);
        tputs(" ");
        tmove(10, 36);
        tputs(m[c].s);
    }
}
```

You can chase the highlight back and forth across the selections, forward with the space bar and backward with the backspace key. Pressing the RETURN or ENTER key will select whichever entry is highlighted when you press the key. You can also make a selection by pressing the key that represents the first character in any entry, regardless of where the highlight is located. Upper and lower case are not the same when menuf() is looking for an initial character. Sometimes it's hard to come up with a list of good descriptive entries that do not have identical initial characters. Distinguishing between cases helps that problem a little. When it really gets tough, you can add leading sequential letters or digits like 'a -' or '7 -' to your entries.

Listing 3, menuf.c, describes the menuf function. It requires the information contained in the header file, hence the

```
#include <menuf.h>
compiler preprocessor directive.
```

```
menuf(m)
struct menus m[];
```

The single argument to menuf is declared to be an array of structures of the type 'menus'.

```
int c, i, j;
int bot, top;
```

Aside from the function argument, the rest of the declarations of variables are integers. 'c' stores values returned from the keyboard, 'i' will keep track of which menu entry is highlighted. 'j' is used twice, first to write the initial display and later to

hold the index of entries while menuf looks for an initial character match. 'bot' and 'top' copy some values from the structure array to make the code more readable.

```
i = bot = m[0].y;
top = m[0].x;
```

'i' and 'bot' are both set to the 'y' value and 'top' is set to the 'x' value in the 0 element of the structure array. 'i' and 'bot' now have the index of the first menu entry and 'top' has the index of the final entry.

```
tmove(m[bot].y, m[bot].x);
hilite(m[bot].s);
```

To write the first menu entry, the tmove function puts the cursor at the coordinates provided in element 1 of the structure array and the hilite function writes the string from that same array element in reverse video.

```
for (j = bot + 1; j <= top; ++j) {
    tmove(m[j].y, m[j].x);
    tputs(m[j].s);
}
```

A 'for' loop beginning with 'bot + 1' steps through the rest of the array to write the remaining menu entries on the screen. tputs writes the strings in normal video.

```
while (1) {
```

A potentially infinite loop started here keeps going around until the user makes a choice.

```
c = getch();
```

Getch waits for a key to be struck and returns the ASCII code for that key. The value that lands in 'c' could represent a menu selection or it could be a signal to

```

/*      menuf.c      Listing 3
*/

#include <menuf.h>
menuf(m)      /* write menu, wait for and return selection */
struct menus m[];
{
    int c, i, j;
    int bot, top;

    i = bot = m[0].y;
    top = m[0].x;
    tmove(m[bot].y, m[bot].x);
    hilite(m[bot].s);
    for (j = bot + 1; j <= top; ++j) {
        tmove(m[j].y, m[j].x);
        tputs(m[j].s);
    }
    while (1) {
        c = getch();
        for (j = bot; j <= top; ++j) if (c == *(m[j].s)) return j;
        tmove(m[i].y, m[i].x);
        tputs(m[i].s);
        switch (c) {
            case '\n': case '\r': return i;
            case FORE: if (++i > top) i = bot; break;
            case BACK: if (--i < bot) i = top;
        }
        tmove(m[i].y, m[i].x);
        hilite(m[i].s);
    }
}

```

move the highlight to another choice. Eco_C's library includes a getch function; C/80's does not. Two versions of getch

are given here--one for C/PM in listing 6 and one for HDOS in listing 7.

for (j = bot; j <= top; ++j) if (c == *

m[j].s)) return j;

Each of the strings in the array is looked at in turn, by incrementing j, and if the first character in the string matches the character typed at the keyboard, the function returns the current value of 'j' as the the number of the menu selection. *m[j].s means the content of the byte at address m[j].s.

```

tmove(m[i].y, m[i].x);
tputs(m[i].s);

```

At this point, the string indexed by 'i' is highlighted because it represents the choice that will be made if the user presses ENTER or RETURN. But we're about to change that--either by signaling for a highlight move, or by making a selection and going somewhere else as directed by the main program. So... this pair of lines rewrites the current menu entry in normal video.

```
switch (c) {
```

Now let's find out what the user wants.

```
case '\n': case '\r': return i;
```

If 'c' is either a newline character or a carriage return, the function returns 'i' as the number of the menu selection.

```
case FORE: if (++i > top) i = bot; break;
```

FORE is the value of the key that moves the highlight forward through the menu so 'i' is incremented before it gets tested by the 'if' statement (++i). If the

VOICE MASTER KEY[®] **VOICE RECOGNITION SYSTEM** FOR PC/COMPATIBLES & TANDY 1000 SERIES A FULL FEATURED VOICE I/O SYSTEM

GIVE A NEW DIMENSION TO PERSONAL COMPUTING. ...The amazing **Voice Master Key System** adds recognition to just about any program or application. Voice command up to 256 keyboard macros from within CAD, desktop publishing, word processing, spread sheet, or game programs. Fully TSR and occupies less than 64K. Instant response time and high recognition accuracy. Voice recognition tool-box utilities are included. **A genuine productivity enhancer!**

SPEECH RECORDING SOFTWARE. ...Digitally record your own speech, sound, or music to put into your own software programs. Software provides sampling rate variations, graphics-based editing, and data compression utilities. Create software sound files you can add to macros for voice recognition verification response. **a complete, superior speech and sound development tool.**

SOFTWARE CONVERSION CODES. ...The **Voice Master Key System** operates a growing list of third party talking software titles using synthesized phonetics (text - to - speech) or digitized PCM, ADPCM, and CVSDM encoded sound files. **Voice Master Key system does it all!**

EVERYTHING INCLUDED. **Voice Master Key System** consists of a plug-in card, durable lightweight microphone headset, software, and manual. Card fits any available slot. External ports consist of mic inputs and volume controlled output sockets. **High quality throughout, easy and fun to use.**

**ONLY \$149.95
COMPLETE**

ORDER HOTLINE: (503) 342-1271 Monday-Friday, 8AM to 5 PM Pacific Time

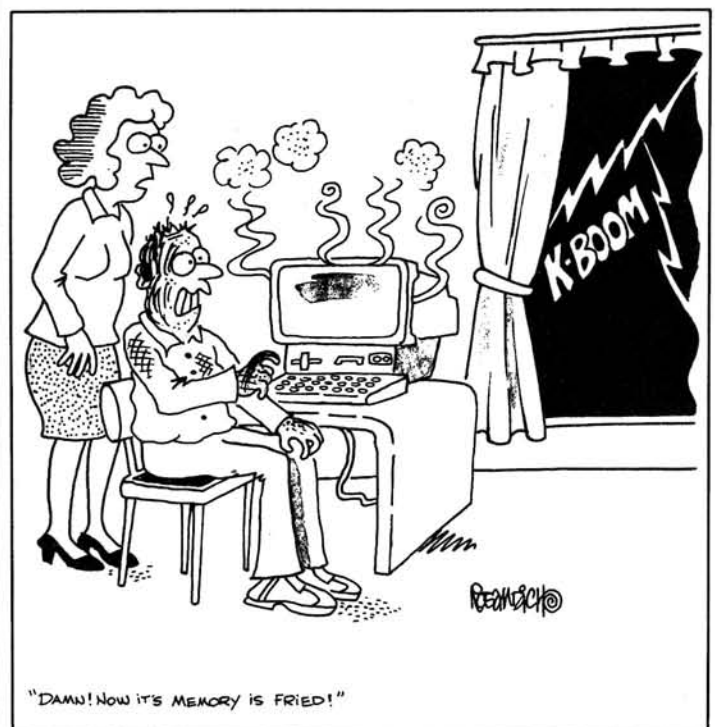
Visa/MasterCard, company checks, money orders, CODs (with prior approval) accepted. Personal checks subject to 3 week shipping delay. Specify computer type and disk format (3 1/2" or 5 1/4") when ordering. Add \$5 shipping charge for delivery in USA and Canada. Foreign inquiries contact Covox for C & F quotes. **30 DAY MONEY BACK GUARANTEE IF NOT COMPLETELY SATISFIED. ONE YEAR WARRANTY ON HARDWARE.**

CALL OR WRITE FOR FREE PRODUCT CATALOG



COVOX INC.
 675 Conger Street, Eugene, OR 97402
 TEL: 503-342-1271 • FAX: 503-342-1283

Reader Service #196



```

/*
*/
/* uses 'stderr' */
tclear() /* clear terminal screen, home cursor */
{
    tputs("\033E");
}
curoff() /* turn off terminal cursor */
{
    tputs("\033x5");
}
curon() /* turn on terminal cursor */
{
    tputs("\033y5");
}
twrap() /* wrap around at end of terminal line */
{
    tputs("\033v");
}
notwrap() /* discard at end of terminal line */
{
    tputs("\033w");
}
tmove(y, x) /* move terminal cursor to line y, column x */
int y, x;
{
    /* 0 <= y <= 23 */
    /* 0 <= x <= 79 */
    putc('\33', stderr);
    putc('Y', stderr);
    putc(y + 32, stderr);
    putc(x + 32, stderr);
}
hilite(s) /* write string s in reverse video */
char *s;
{
    tputs("\033p");
    tputs(s);
    tputs("\033q");
}
tputs(s) char *s; { while (*s) putc(*s++, stderr); } /* write s to screen */

```

getch.cpm Listing 6

```

/*
*/
getch() /* read single character from console (C/80, CP/M) */
{
    int c;
    while ((c = bdos(6, 0xFF)) == 0);
    return c;
}
#endif

```

```

tputs(s) char *s; { while (*s) putc(*s++, stderr); } /* write s to screen */

```

```

/*
*/
minicurs.h19 Listing 5 H19 (or H89) terminal control

```

```

/*      getch.hos      Listing 7
*/

getch()      /* read single character from console (C/80, HDOS) */
{
#asm
    LXI      B,0          /* get          */
    XRA      A            /* current    */
    DB       255,6        /* console mode */
    MOV      B,A          /* save       */
    MOV      C,A          /* current mode */
    PUSH     B            /* on stack   */
    MVI      B,133        /* specify    */
    MVI      C,133        /* raw input, */
    XRA      A            /* no echo,   */
    DB       255,6        /* character mode /
    DB       255,1        /* read console and
    JC       $-2          /* loop til a key is struck */
    POP      B            /* get entry mode back */
    MVI      H,0          /* save       */
    MOV      L,A          /* character  */
    PUSH     H            /* on stack   */
    XRA      A            /* restore    */
    DB       255,6        /* entry mode */
    POP      H            /* get character back for getch to return */
#endasm
}

```

incremented 'i' turns out to be higher than the highest index number in the menu array (top) 'i' is set to the value of the lowest index number (bot). That way, when the the highlight comes to the end of the menu, pressing the FORE key again makes it start over at the beginning. The switch test knows what key was pressed now, and the break statement terminates the process of looking at cases.

case BACK: if (--i < bot) i = top;

Much like the previous case but with the highlight moving in the other direction. 'i' is decremented before it's tested and if it's lower than the first entry's index it gets set to the last entry's index. There is no break statement here because execution is about to fall out of the switch test anyway.

Between these two cases, the user can go around and around the menu choices in either direction.

```

tmove(m[i].y, m[i].x);
hilite(m[i].s);

```

The highlight is written at its new location. Actually, it might not be a new location. If the user hit an undefined key, the highlight is rewritten in its old position--in effect, no change.

```

}

```

The first curly brace ends the 'while' loop and the other one closes the function. The 'while' loop will be reiterated if the switch test found a highlight move. menuf returns only if a menu entry's initial character key was pressed or if the user hit ENTER or RETURN.

Listings 4 through 7 supply support functions for menuf() and for programs that use menuf().

Minicurs.ans in listing 4 is for a system

that's using an ANSI device driver for the terminal or any terminal that recognizes ANSI codes. Minicurs.h19 in listing 5 is for the H19 terminal, H89 computer or any terminal using Heath terminal control codes. Whichever version of minicurs is used, function names and arguments are the same. Only the control strings written by the functions differ. A program that calls functions in minicurs doesn't care which version it gets.

Tclear() clears the terminal screen and places the cursor--visible or not-- in the upper left corner of the screen.

Curoff() makes the terminal cursor invisible and curon() makes it visible.

There are no arguments to tclear(), curoff() or curon().

If your terminal is configured to wrap around at the end of the line, writing strings that are not terminated by a newline to the screen will result in scrambling the display. The terminal counts characters -- including control characters - and when it has processed eighty of them, it writes a newline to the screen and a string you intended for a specific location suddenly breaks and continues at the beginning of the next line. Several of those events will cause your display to start scrolling off the screen, too. You can put a call to notwrap() in your program before you start creating menu screens if you need it. That will cause the terminal to discard characters at the end of a line instead of issuing disastrous newlines. Then you should put a call to twrap() just prior to the program's exit() to restore wraparound. There are no arguments to twrap() or notwrap().

Tmove(y, x) moves the terminal cursor to line 'y' and column 'x'. Lines are numbered from 0 at the top to 23 at the

bottom. Columns are numbered from 0 at the left to 79 at the right.

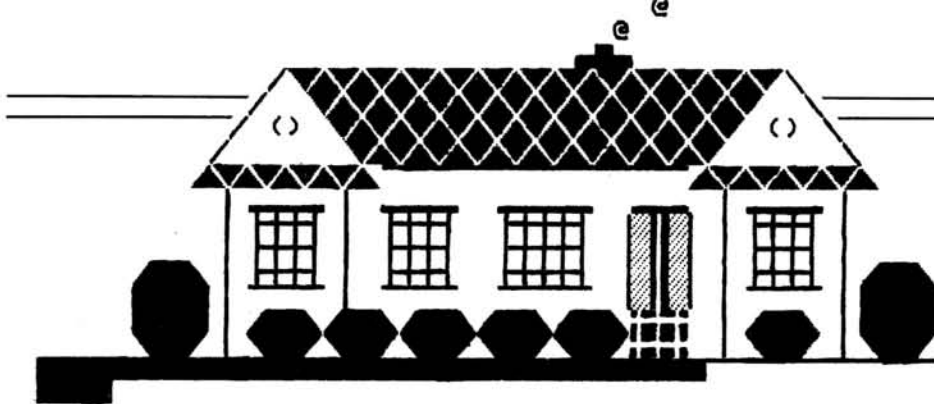
Hilite(s) writes a string--pointed to by the argument 's'--in reverse video to the terminal screen, starting wherever the cursor is at the time of the call to hilite.

Tputs(s) writes a string--pointed to by the argument 's'--in normal video to the terminal screen, starting wherever the cursor is at the time of the call to tputs. Tputs requires the putc function from the C library.

Eco__C supplies a getch() function to return a single character from the keyboard without waiting for a carriage return and without echoing the character to the screen. C/80 has no such function so two versions are given here: getch.cpm in listing 6 for the CP/M operating system and getch.hos in listing 7 for HDOS. There are no arguments to getch. The return value of the function is the ASCII value of the key struck. Getch.cpm uses the C/80 bdos system call to do direct console input. Getch.hos is written in 8080 assembly language to use HDOS system calls, even though the word 'SCALL' doesn't appear anywhere in the source file. A statement like 'SCALL .SCIN' is replaced by 'DB 255,1' to make the code readable by both AS.ABS (the assembler distributed with C/80) and Macro-80 (the relocating macro assembler from Microsoft). Both versions of getch() do raw character input so no special processing is done for control characters. CTL-C for instance would just return a value of 3.

There are far more elaborate schemes for producing highlight oriented menus. This one is fairly quick and doesn't use a lot of space. It would be possible but cumbersome to add a help screen call for each menu entry. The difficulty would come in rewriting the screen as it was before the help screen call. But this system works without window or graphics libraries, which is why its system demands are modest. *

**EXPLORE
NEW WORLDS
WITH
HUG
GAME
SOFTWARE**



GREETINGS &
SALUTATIONS FROM
THE GREAT
DISMAL SWAMP!

A>

Dress Up the CP/M 'A>' Prompt With a Picture!

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

Tired of the cold, plain "A>" prompt after booting? Upon a WARM or COLD boot, you can now display any opening screen you design, such as the one above. Using "PAINT.ASM" and the enclosed program, "HELLO.ASM", the process is quick, and fun.

Background

Before we begin, let me quickly update you on the status of "PAINT.ASM".

As some of you may recall, the March and April 1986 issues of "REMark" contained my first articles on "PAINT.ASM". The articles described an H/Z-89/90 CP/M program which permitted the drawing of a graphics screen or picture, in a relatively simple process.

Upon completion of the "picture", it could be loaded into computer memory, saved to disk, or dumped to a dot matrix, near letter quality, multi-mode printer, such as the Panasonic KX-P1092. The program also permitted the loading of special character fonts, developed by us and easily modified, to use with the printer.

Other enhancements were described in a follow-up article, "Enhanced PAINT.ASM!!!", published in the February 1987 issue of "REMark".

That summer, I was transferred from Columbia, MD, to Elizabeth City, NC. Needless to say all hobby use of my computers ceased and I apologize to those who wrote inquiring about an update.

I have finally completed my move, so please note the changed address above. It's hard to believe how quickly time has flown — spent a year fixing up the house, learning my new job, and pumping out of-office work. Only recently, have I begun playing with "PAINT" again, completely rewriting portions as enhancements were made. I also added:

1. A printer tabbing routine to move the printed picture horizontally anywhere on the paper, such as positioning the picture in the lower-right quadrant for making greeting cards.
2. A routine to permit the display of the opening screen of instructions at any time, without destroying the current work.
3. Better printer control — enhanced/expanded line spacing routine, compressed printing (132 CPL), through a second menu screen.
4. A routine to eliminate the color escape codes from screens made by the H/Z-100 (not PC) version, giving some portability of pictures from one machine to the other. The H/Z-100 has no problem displaying H/Z-89 pictures, though without color, of course.

Additionally, and after long last, a color version is complete for the H/Z-100.

Up until now, I've only seen color done as bit graphics: draw a line, arc, circle, box, etc., then color it in. I'd found these programs highly sophisticated and complex, much more than I needed.

The H/Z-100 program only uses the standard H/Z-89 graphics set, in eight colors, then saves them to memory or disk in precisely the same manner as the H/Z-89 version.

Color is added through an ESC-'m', followed by two numbers indicating the foreground and background color, immediately preceding each character. As discussed in the last article, when we wanted to "MOVE" or "COPY" blocks of reverse video or graphics characters, it complicated those routines tremendously — you had to take the color code with the block, but also leave it behind if that color was to continue with characters to the right of the block.

Make sense? (I'm not sure I understood it.) In any case, I can't cover the whole process here, but I will try to explain it in detail in another article. Both programs are now available and I'll finish the article with information on how to obtain them.

One sour note to the whole thing — Printer routines. As I've explained before, printer codes are not standardized. To those who write with problems getting their printers to work with these routines, I can only study the documentation they send and suggest possible remedies and modifications. I have no experience with any other printers but mine. In one case, I've even completely modified the program to use 8-pin vice 9-pin graphics and mailed it out, with my best wishes.

At the outset, I said these articles were to encourage you to dig into Assembly Language and modify the program as necessary to meet your own needs. Judging from the letters I've received, many of you have been doing just that, and I congratulate you. It's not easy, but I think you'll agree that it's rewarding getting something you programmed to work, especially when it's something useful.

Enough! Let's get to another, but much smaller project.

Boot Screens — In Color Even!!

If you recall, in the previous articles I gave several suggestions on the use of "PAINT", including making and printing letterhead and greeting cards, graphics screens for games, and just plain drawing fun for youngsters. The program has been working great in doing all of these, especially the screens made up for inclusion in games, which add a lot more zip and interest.

Well, I've been wanting a quick

method to place an opening screen on the computer for years — just like the “compatibles”. I’ve finally got it! It uses any version of the “PAINT” programs, as long as the pictures could be saved to disk.

The new program does nothing more than call up and display one of these previously made pictures. Therefore, if you’ve experimented making these pictures, you know how easy it is. Take one of those you made for a letterhead, add a

few words of greeting, and it can easily be displayed during a Boot.

Different opening screens can be made for each season, holiday or occasion. I’m taking a Christmas scene I made last year and will use that during December. Swapping screens is a simple matter of renaming files.

For the H/Z-100, I simply take the older H/Z-89 screens, and retype those portions I want in color — that simple.

To those who worked on the pre-

vious “PAINT” programs, the routines of the following program are pretty familiar, with minor changes. We again make use of preprinted messages stored toward the end of the program. The messages are printed, using the “PMSG” routine, until a delimiter, ‘\$’, is found.

As with most programs, the “START” routine saves CP/M’s Stack and Flags so we can return there when our program is done. “MSG1” clears the screen, in preparation to receiving our screen of data.

Listing for “HELLO.ASM”

; VAGTS “HELLO” PROGRAM - 10/4/88

; Mr. Steven W. Vagts

; 2409 Riddick Rd

; Elizabeth City, NC 27909

; (919) 335-7487

;

; Following codes are from CP/M INTERFACE GUIDE

CONOUT EQU 2 ;CONSOLE OUTPUT

PRINT EQU 9 ;PRINT STRING TO SCREEN

OPEN EQU 15 ;OPEN FILE

CLOSE EQU 16 ;CLOSE FILE

RDFIL EQU 20 ;READ FILE (SEQUENTIAL)

BDOS EQU 5 ;BDOS VECTOR

DFCB EQU 05CH ;DEFAULT FCB

DMA EQU 080H ;DMA AREA

ESCAPE EQU 27 ;ESC CHAR

ORG 100H

START EQU \$

LXI H,0 ;HL REG=0

DAD SP ;LOCATE STACK

LXI SP,STACK ;SET OUR OWN

PUSH H ;SAVE CP/M S STACK

PUSH PSW ;SAVE CP/M S FLAGS

LXI D,MSG1 ;CLEAR SCREEN

CALL PMSG

CLRFCB LXI D,DFCB+1 ;BEGIN OF FCB

MVI B,31

MVI A,' '

CLEAR STAX D ;CLEAR FCB

INX D

DCR B

JNZ CLEAR

XRA A

STA DFCB ;0 DRIVE CODE

STA DFCB+12 ;0 EXTENT

STA DFCB+13

STA DFCB+14

STA DFCB+15 ;0 RECORD COUNT

STA DFCB+32 ;0 CURRENT RECORD

READFL MVI B,11

LXI D,DFCB+1 ;POINT TO BEGIN FCB

LXI H,PSCRN1 ;FILE NAME ADDR

RDNAME MOV A,M

STAX D

INX D

INX H

DCR B

JNZ RDNAME

OPNFIL MVI C,OPEN

LXI D,DFCB

CALL BDOS ;TRY TO OPEN FILE

CPI 255 ;A=255 IF ERROR

JZ NOFIL ;OPEN FAILED

RDREC LXI D,DFCB ;FCB ADDRESS

MVI C,RDFIL

CALL BDOS ;READ A RECORD

ORA A ;READ OK?

JNZ NOREAD ;UNSUCCESSFUL READ

LXI H,DMA ;POINT TO DMA AREA

MVI B,128 ;PRINT 128 CHARS

RDLOOP MOV A,M ;GET CHAR

CPI 'Z'-40H ;CTRL-Z?

JZ RDDONE ;READ DONE

CALL CHOUT

INX H

DCR B

JNZ RDLOOP ;LOOP UNTIL 128 CHARS PRINTED

JMP RDREC

CHOUT PUSH H

PUSH B

MOV E,M ;E=CHAR

MVI C,CONOUT

CALL BDOS

POP B

POP H

RET

RDDONE CALL CLOSDK ;READ DONE

LXI D,MSG5

CALL PMSG

JMP EXIT1

NOFIL LXI D,MSG2 ;HELLO.SCN NOT ON DISK

CALL PMSG

JMP EXIT

NOREAD LXI D,MSG3 ;UNSUCCESSFUL READ

CALL PMSG

CALL CLOSDK

LXI D,MSG4

CALL PMSG

EXIT1 POP PSW ;GET CP/M FLAGS

POP H ;GET CP/M STACK

SPHL ;SET IT

RET ;RETURN TO CP/M

CLOSDK MVI C,CLOSE

LXI D,DFCB

CALL BDOS ;CLOSE FILE

RET

PMSG PUSH PSW

PUSH H

MVI C,PRINT

CALL BDOS

POP H

POP PSW

RET

MSG1 DB 27,72,27,121,49,27,69,27,106,27,120,53,'\$'

MSG2 DB 'HELLO.SCN WAS NOT FOUND\$'

MSG3 DB 'UNSUCCESSFUL READ OPERATION!\$'

MSG4 DB 27,72,27,121,49,27,121,53,\$

MSG5 DB 27,89,54,37,27,121,49,27,121,53,'\$'

PSCRN1 DB 'HELLO SCN'

DS 32 ;SPACE FOR STACK

EQU \$;PUT STACK HERE

END START

"CLRFCB" clears the File Control Block, which permits us to read/write files from/to a storage device (the disk). Whenever we specify a file name for disk operations, the FCB is used. A complete discussion of this was done in my March 1986 article.

After clearing the FCB, "READFL" places the name of our data file (HELLO.SCN) into it so we can attempt to open the file ("OPNFIL"). The file name is defined as variable PSCRN1 at the end of the program. For flexibility, "READFL" requires the file name consist of the full eleven characters, the last three of which give the file type. The decimal between the file name and file type is not used and spaces are added to make the name the required length. Defining PSCRN1 in this manner permits using any CP/M legal file name for our screen, if necessary.

Once the defined file is opened, we read the data sequentially, a record at a time ("RDREC"). A 128 byte record is read from the file and placed into memory at the current DMA (Direct Memory Address), a temporary holding area for data being read from or written to a disk.

The read loop ("RDLLOOP") then takes each of the characters, and prints them on the screen ("CHOUT") until all 128 characters are done or until a "CTRL-Z" is found, which is the end-of-file marker. Until a "CTRL-Z" is found, successive records are read and displayed on the screen.

Completing the reading of the disk file, we must close ("RDDONE" & "CLOSDK") the file we had opened and reset our cursor to a desired position on the screen, in this case, line 23 ("MSG5") on the screen.

Line 23 is used rather than line 24 because following the printing of "MSG5", CP/M's "PRINT" function gives a carriage return which takes the cursor to line 24 anyway. If the cursor were on line 24 initially, the whole screen would scroll up one line when this was done.

"NOFIL" and "NOREAD" provide error messages, "MSG2" & "MSG3", if "HELLO.SCN" (our data file) was not on the disk or if the read was unsuccessful.

Finally, "EXIT" recalls the CP/M flags and stack, returns us to the CP/M environment and "A>" prompt, and is ready for your first command.

After typing the program using your favorite editor or word processing package, assemble the program, which creates "HELLO.HEX", using CP/M's "ASM" command and use the "LOAD" command to create the "HELLO.COM" file. Instructions are given in the CP/M Guide.

Rename one of your previous "PAINT" data files to "HELLO.SCN" and type "HELLO" at the "A>" prompt to check the program. The screen should clear and the picture appear with the "A>" relocated to line 24.

Reconfiguring the BIOS

Once this program has been debugged and works properly, you must reconfigure your CP/M's BIOS to run this program automatically upon BOOTing the computer.

For the Heath/Zenith CP/M BIOS, this simply entails running "CONFIGUR" after the "A>" prompt. Type the file name "HELLO" at the appropriate Command Line, for WARM and/or COLD BOOT. The procedures are clearly given in the CP/M manual. Make the changes permanent to memory and disk.

For those of us using C.D.R.'s BIOS for the H/Z-89, run "I/OMOD", which does essentially the same thing, but run this program twice, first typing "1" to make the changes to the BIOS, then run again, this time typing "2" to make the changes to "PUTCPMxx", which, when run, places the system on other disks.

From here on, when doing a warm or cold boot, depending upon how you answered the above questions, the boot will automatically be followed by execution of the program "HELLO". "HELLO" will look for the data file "HELLO.SCN".

Closing

Upon request, I would be happy to send anyone the source code for "PAINT-89", "PAINT100", and "HELLO" for \$8.00. I'll also send the finished products, ASM and COM files to all three programs, on disk for \$15.00, if you include a preformatted disk with your request. Upgrades to prior customers will be \$10.00. I now primarily use H/Z CP/M (H/Z-100) or CDR CP/M (H/Z-89) and double-sided, double-density extended, soft-sectored disks. However, I can still copy to the older 10 sector, single-sided disks. Please include a phone number in case I have problems.

As I've said on numerous occasions, my primary interest is to generate interest in Assembly Language. I'm hoping you will want to experiment and develop your own changes — unless your printer is similar to my Panasonic KX-P1092, you will have to modify the printer routines, as a minimum anyway.

In the process of coding, assembling and modifying these programs, if you have enjoyed the work, sweat, and, yes, some reddening of the eyeballs from the late nights when the gremlins just don't seem to cooperate, and finally, the swelling of the chest with pride when the program finally works properly, then I have completed my goal.

Please feel free to distribute these programs to your friends as you see fit.

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



Back to the Books

Let's face it, sooner or later you're gonna have to try and read those computer USER manuals! But, before you do, read "POWERING UP". This book was written especially for you in a non-technical, easy-to-understand style. Who knows, with "POWERING UP", you may NEVER have to read your user's manuals again! Order HUG P/N 885-4604 today!



Why didn't somebody think of this sooner! Books and magazines are printed on both sides of the page, why not listings from your computer's printer? With HUG's "Both Sides" printer utility, now you can send your printer's output to both sides of the page; properly positioned, and page numbered! Cut your paper usage, and binder space usage in half! Order HUG's "Both Sides" printer utility today. **Both Sides** is available for MSDOS and is HUG P/N 885-3048.

Beginner's Batches

Mark Haverstock
6835 Colleen Drive
Youngstown, OH 44512

Whip Up a Few Batch Files to Automatically Handle DOS Functions

Is your Heathkit/Zenith computer really saving you time? Do you find yourself bogged down at the system prompt repeatedly typing the same DOS instructions? MS-DOS provides a powerful, but often neglected, feature called a batch file. It can save time by automatically carrying out these routine functions, and will even load your favorite program without touching the keyboard.

What is a batch file? It's simply a series of ordinary DOS commands which are written in a text file. Any DOS command that you'd normally type in at the A> prompt, such as DIR or COPY, can be included in a batch file. When you write a batch file, you make a list of instructions for the computer to carry out. In effect, it makes it possible for the computer to run on "auto pilot", carrying out the commands in sequence, just as if you'd typed them in yourself.

One kind of batch file that is particularly useful is the AUTOEXEC.BAT, which operates when you turn on your computer. Let's take a look at some examples of AUTOEXEC.BAT files.

Taking it From the Top

Whenever you turn on your computer, it automatically looks in the root directory for a file named AUTOEXEC.BAT on the startup disk. If this file is present, it carries out the DOS commands found in the file.

We'll start with a simple example. If you have a floppy disk system, you proba-

bly boot up with your MS-DOS disk (or one containing COMMAND.COM) in drive A. Suppose you want to see a directory of files on drive A. At the DOS prompt, you'd probably type:

```
DIR/W
```

The screen would then display the directory in column format across the screen.

This same command could become a one-line batch file. On boot up, your computer will load DOS as usual. Next, it will automatically display the directory. The results would be the same as if you typed in the DIR/W yourself. If you watch closely, you'll see the DIR/W appear as the batch file runs.

Short AUTOEXEC.BAT files like these can easily be created with the DOS COPY command directly from the keyboard. The batch file just described would look something like this:

```
COPY CON AUTOEXEC.BAT
DIR/W
^Z
```

COPY CON stands for COPY from CONsole (or keyboard). Each command to be executed, such as DIR/W, should appear on a separate line followed by a carriage return. The ^Z marks the end of the file and is produced by pressing the F6 function key.

Starting Your Programs Automatically

The next sample batch file is designed to run a program automatically when you start up your computer. In this example, let's assume that you want to automatically run a data base program named dBASE. The batch file would be

written like this:

```
COPY CON AUTOEXEC.BAT
ECHO OFF
CLS
DBASE
^Z
```

I added the ECHO OFF and CLS commands for neatness sake. ECHO OFF tells DOS not to display the commands that follow as they are executed. CLS clears the screen, so only the program will be displayed when it is loaded. dBASE, of course, is the program we're going to run. You can substitute the name of any program you want to run in place of dBASE.

The next example is a variation of the batch file listed above. Let's suppose the program dBASE is located in a subdirectory named DB. To run the program dBASE, the computer first has to change to the subdirectory DB, and call the data base program named dBASE. Here's how it looks:

```
COPY CON AUTOEXEC.BAT
ECHO OFF
CLS
CD\DB
DBASE
^Z
```

It is important to remember that an AUTOEXEC.BAT file must be stored in the root directory if it is to function when the computer starts. It will be ignored if stored in any other directory.

Starting Utility Programs Automatically

Many PC-compatible owners have utility programs that they would like to load in automatically, without the bother of executing each at the DOS prompt. For example, if you have a clock card and a mouse, you may want to have these up and running before you move into a program. The next example loads drivers for

both the mouse and clock, as well as displaying the directory at the end.

```
COPY CON AUTOEXEC.BAT
ECHO OFF
CLS
CLOCK
MOUSE
ECHO ON
DIR/W
^Z
```

The ECHO OFF and CLS again clear the screen as the programs CLOCK and MOUSE are loaded. ECHO ON returns the screen display so the directory can be viewed. If you also have a menu program, you may want to substitute the name of the program in place of DIR/W.

Batch Files in Subdirectories

Sometimes it is useful to have batch files to do specific tasks within a program directory. Unlike the AUTOEXEC.BAT files shown earlier, this one is called only when needed. It is a batch file I often include in word processors and data bases simply to make backups more convenient. Once I've exited the main program, all I need to do is to type SAVE at the DOS prompt, and it automatically begins the backup, prompting along the way. This example, SAVE.BAT, backs up data base files with the extension .DB from drive C, a hard drive, to drive A.

```
COPY CON SAVE.BAT
ECHO OFF
CLS
ECHO INSERT BACKUP DISK IN DRIVE A AND CLOSE DRIVE DOOR
PAUSE
CLS
ECHO COPYING FILES
COPY C:*.DB A:
CLS
ECHO DONE COPYING FILES. REMOVE DISK FROM DRIVE A AND
ECHO LABEL "BACKUP DISK" WITH TODAY'S DATE
^Z
```

Here's an explanation of how it works: ECHO OFF and CLS keep DOS from displaying commands on the screen. ECHO prints a prompt I want to appear on the screen. ECHO works on a line-by-line basis, without affecting the ECHO OFF before it.

PAUSE prints "Strike any key to continue" on the screen and waits for a key to be pressed before moving on to the next step. This is handy, because it gives you time to insert the disk, and check to see that everything is ready before proceeding.

The next echo prints the current status of the program on the screen. It is in the process of copying the files. COPY C:*.DB is the COPY command from DOS that is actually performing the work. It is

using a wildcard copy to copy all data files, the ones with the .DB extension, from drive C to drive A.

CLS clears the screen to prepare for the directions that will be printed with the last ECHO.

In conclusion, simple batch files such as these can be a handy addition to your start-up routine, and can provide some time-saving shortcuts. The ones in this article are only a few examples of useful batch files. Experiment a little and write your own "script" for DOS! Remember, any command that can be entered at the system prompt may be included in batch files. *

Tips on Batch Files

Using AUTOEXEC.BAT to Load Memory-Resident Programs

Some memory-resident programs when loaded with others, such as SIDEKICK, must be loaded in a certain order. Be sure to check each program's documentation for loading instructions before writing the batch file. If a batch file doesn't work properly, try rearranging the order in which the programs load.

ECHO Command

Be sure to leave a space between the ECHO command and the text which follows. Avoid using characters such as < and > because they will cause an error message.

Creating and Naming Batch Files

Longer batch files may be created more easily with a word processor. This can make the process of editing and test-

ing much easier. However, the batch file must be saved in ASCII format. Refer to your word processor manual for instructions on how to do this.

Batch files can be saved under almost any name, as long as they conform with DOS naming rules and end with a .BAT extension. Be careful not to name batch files after DOS commands. For example, if you named one DIR.BAT, DOS would not execute the batch file. Instead, it would assume you wanted the DIRectory command.

Time and Date Prompts

Unless time and date prompts are written into batch files, they're bypassed. If you have a clock card, add the name of the clock driver program (usually CLOCK) to the batch file. For entering time manually, add the commands TIME and DATE to your batch file. *

Hard Drive and Expanded Memory Support Changes in Zenith MS-DOS 3.3 Plus

William M. Adney

P.O. Box 531655

Grand Prairie, TX 75053-1655

Copyright © 1989 by William M. Adney. All rights reserved.

Your response to the Powering Up book continues to be overwhelming, and it makes me wonder what I will do for an encore. In retrospect, the need for that kind of publication is obvious, and I can't help but wonder why I did not think of it before. Actually, the project was generally suggested by many of you who needed some beginning information on Heath and Zenith PC compatibles, and this particular book has already been so successful that I am considering some additional similar projects. Although all of my articles have generally been intended to help you learn about your computer in one way or another, there are a couple of possibilities for additional projects that come to mind.

Powering Up (To Be Continued . . .)

Perhaps the most obvious project is, of course, Powering Up — Volume 2, which would include additional topics that are more detailed and are specific to Heath and Zenith PC compatible computers. Assuming you know the basic information included in the original Powering Up, Volume 2 would include topics such as what to do when you see the old "Abort, Retry, Ignore" message, ROM error messages, how to use Function Keys to edit command lines, how to use EDLIN to create and edit files, computer numbering systems (e.g., hex), and similar topics. Volume 2 would also include the latest information on DOS commands. Zenith has recently released MS-DOS version 3.3 Plus which has some new features, especially for hard disk users. If you have any ideas for topics, be sure to let me know right away because I am currently in the process of defining the topics for this new book. The topics should be general in scope and should apply to most, if not all, of the Heath/Zenith PC compatible

computers. Because of the current interest in laptop computers, I may even include a general article on laptops because I now have a SupersPort 286. These are just a few of my own ideas, and I would appreciate hearing from you if you have others. But speaking of MS-DOS 3.3 Plus...

Zenith MS-DOS 3.3 Plus

Although I have not written before about Zenith's new release of MS-DOS 3.3 Plus, I have already received some questions about it. If you have been faithful in sending in your registration cards, you have probably received the update notice about obtaining this version. If you purchased a system in the last six months, however, I understand that ZDS is trying to get caught up on software registration, so there is a chance you may not have received the update notice yet. Sounds like Zenith may have some problems in getting the information entered and updated on a computer! Computers are good tools, but they do require constant feeding.

Several people have called to tell me that MS-DOS 3.3 Plus is "not compatible" with some utility programs, notably Mace Utilities and Norton Utilities. Actually, the reverse is true. It turns out that virtually all of the older disk utility programs are not compatible with both Zenith MS-DOS 3.3 Plus and IBM PC-DOS release 4.0. Although that may seem to be bandying words about and hedging the issue, Microsoft has made some fundamental technical changes in the latest DOS versions that are only hinted at on the update card for MS-DOS 3.3 Plus. To understand why older versions of the Mace Utilities and Norton Utilities will not work with this new DOS release, let's examine

some of the hints given on the Zenith update card and then take a quick look at what some of these technical changes are.

The update card notes that the new MS-DOS 3.3 Plus "supports drives larger than 32 MB" (no big change here from the previous release, but I think that word "drives" should have been "partitions"), but also mentions that it "allows partitioning of disks larger than 128 MB." Assuming that my guess is correct, one thing you must know is that previous versions of Zenith MS-DOS supported a maximum of 32 MB per partition and four partitions per physical drive. That gives you the maximum possible physical drive capacity of 128 MB. By the way, this feature was only available in PC compatible MS-DOS releases and does not apply to the Z-100. Also, IBM added the partitioning feature to the FDISK command in current version 4.0 PC-DOS releases, but there have been many reported bugs in these versions.

For those of you interested in the technical details, the ability to use larger hard disks was added by using larger cluster factors, among other things. If you have a large enough hard disk, the PART command will now display two additional values in the Partition Type — "DOS (BIG)" or "DOS(EXT)" — in addition to the "DOS(12)" (12-bit FAT entries) and "DOS (16)" (16-bit FAT entries) values. I have already done some experimenting with this since I recently implemented an 80 MB Seagate ST-4096 hard drive on my Z-386 with MS-DOS 3.3 Plus, and that will prob-

ably be a separate article for *REMark* because of length. The PREP command has been changed slightly to only make four passes, and it now displays what it is doing (write or verify) by head and cylinder number, similar to *FORMAT*. This is a nice feature because you can tell where PREP is and estimate how long it will take to finish the low-level format. PART has been changed considerably, and it now uses pop-up windows with color. The enhanced CONFIGUR command also uses pop-up windows and displays in color, too.

Programs that require direct access to the disk — disk optimizers, undelete/unerase programs, and sector level disk editors, such as HUG's HADES — typically use a special feature called Interrupts or INTs. Because of the expansion in partition and physical disk sizes in Zenith MS-DOS 3.3 Plus and IBM PC-DOS 4.0, two interrupts had to have a technical enhancement to cope with these larger sizes. For programmers, this has a significant impact on direct disk access because the coding scheme for INT 25h (Absolute Disk Read) and INT 26h (Absolute Disk Write) changed. For both interrupts, the CX register is now set to -1 for partitions larger than 32 MB, instead of containing the number of sectors to read or write. In addition, DS:BX also contains a pointer to a parameter block for partitions larger than 32 MB where the previous call used DS:BX as a pointer to a buffer for the disk data. There are other technical changes too, but these particular changes will probably (hopefully!) cause most older programs to fail.

Since these programs are quite specific in their direct disk access requirements, older versions of the Mace Utilities and Norton Utilities will not work correctly for large partitions with Zenith MS-DOS 3.3 Plus or IBM PC-DOS 4.0 because of the technical changes in these latest DOS versions. Although you will probably find that none of the older disk-level programs will work at all with these new DOS releases, DO NOT even try to use them — you may DESTROY everything on your hard disk and have to start over with the PREP command! If you have no idea what I was talking about in the previous paragraph, just assume I know the subject and don't use these older programs. If you feel compelled to experiment with the new DOS release and any of these kinds of programs, be absolutely sure that you take a complete backup of your hard disk (ALL partitions) before you begin. Any older program that fools around with direct access to the disk, such as a disk optimizer, an unfragment program or an undelete/unerase program, may wipe out data or work incorrectly on a hard disk because of these new changes. However, I have already tried the latest MACE Utilities release, called MACE GOLD, and its

UNFRAG program works just fine with Zenith MS-DOS 3.3 Plus in a 48 MB partition.

As far as I know, Zenith is the only manufacturer that anticipated some of these potential problems because, as noted on the update card, the new MS-DOS 3.3 Plus includes: "Zenith Disk Utilities which allow users the opportunity to recover accidentally deleted files" and the "COMPACT utility which reduces file fragmentation" (i.e., an optimizer/unfragment program). This thoughtful addition to the normal DOS programs by Zenith will help you put off updating at least some of your software if you decide to implement 3.3 Plus.

These technical changes may also affect other programs that perform disk-level access, such as backup/restore programs like Fastback. Many programs of this type may perform direct disk access so that they also cannot be used under these new DOS releases. In addition, I am also anticipating that some application programs, particularly games, will also not run under these new DOS releases because of some copy-protection scheme that performs disk level access. There is not much you can do to fix this kind of problem since it is related to something in the application program.

Zenith has enhanced some of the existing commands, like the PART and FORMAT commands, to work with larger partitions. But the changes for hard drive definition and usage have changed significantly in the new MS-DOS 3.3 Plus, so let's see what happened.

The Primary Partition

There have been some changes in the terminology used to discuss hard disks in the manual, and there are a couple of new terms that you need to know about. I already mentioned that I have recently purchased an 80 MB Seagate ST-4096 drive from Payload Computers for my '386 system, and although I will write a separate article on the installation and implementation of it with version 3.3 Plus, it is worthwhile to discuss some of the old and new terminology here as it applies to the PART command.

In Zenith MS-DOS versions prior to 3.3 Plus, there was really only one kind of partition that you could define with the PART command on a single physical hard drive. In MS-DOS 3.3 Plus, this is now called a PRIMARY partition. For older Zenith MS-DOS versions, all partitions were primary partitions. When you used these older MS-DOS versions, you usually defined one bootable partition and could define up to three "other" partitions with the PART command. And many times it is convenient, but not required, to define the first partition as the bootable partition.

Before I go any further, let me tell you

now that some of this information is a little confusing, especially if you don't know too much about how hard drives work. I will try to take you through this discussion by small steps, and if you find it somewhat baffling, bewildering, and befuddling, consider how much fun I had in trying to write about it. I lost track of how many times I rewrote this particular information. And if you enjoy learning about this new stuff as much as I enjoyed learning and writing about it, be sure to contact your local IBM representative to express your profound appreciation for all of these confusing technical problems and new definitions related to this convention.

The concept of the Primary partition is not really too difficult to understand because that is directly associated with the Partition Table that could be changed to the Zenith PART command or IBM FDISK command. For discussion's sake, and to avoid confusion later on, I will define a new term called the Primary Partition Table. Note that this is a term that I have defined, and it is not found in any DOS manual I have seen. I have found it necessary to define this new term to differentiate it from the Extended Partition Table that I will define later on in this article.

The PRIMARY PARTITION TABLE is defined as a "list" (i.e., table) of up to four hard drive parameters, including the beginning and ending cylinder for each partition, that is initially created in the reserved area of a hard disk by a low-level format program, such as Zenith's PREP command. The single, most important piece of this definition is that the Primary Partition Table was created by a low-level format program, like PREP. This is true for virtually all DOS versions, including Zenith MS-DOS 3.3 Plus and PC-DOS 4.0. The Zenith PART and IBM FDISK commands are used to add and delete partition definitions in the Primary Partition Table. Similarly, a PRIMARY PARTITION is a partition whose entry is contained in the Primary Partition Table. If you are interested in the detailed contents of the Primary Partition Table, most of the Zenith MS-DOS manuals include a list and discussion of the parameters in it as part of the PREP command.

As I mentioned before, all partitions in Zenith MS-DOS prior to version 3.3 Plus were primary partitions that you could change with the PART command. Each partition was limited to a maximum of 32 MB, and up to four partitions could be defined for MS-DOS so that you could use up to a 128 MB hard drive with Zenith MS-DOS versions 3.10, 3.20 or 3.21. In order to use that capacity, you had to change the Partition Assignment Flag (using CONFIGUR or DSKSETUP, depending on the MS-DOS version) from Automatic to Manual, and then you had to use the ASGNPART command to make those partitions available to MS-DOS if you had a

large hard drive beyond the famous 32 MB limit imposed by IBM PC-DOS versions prior to 4.0.

In case you are wondering exactly why the PC compatible DOS versions are limited to a maximum of four partitions, there is really no valid technical reason for it, except that it was an IBM-imposed design limitation like the 640 KB memory limit. Zenith was forced to also use that design limitation in order to be PC compatible, especially with respect to the Primary Partition Table. Since some of you may not be familiar with this statement of fact, let me point out that the Z-100 MS-DOS PART command could define up to 16 Primary Partitions, so there is no real technical reason why four should be the limit, except to maintain compatibility with IBM and prior Zenith MS-DOS releases, of course.

But to really understand why there are some new and changed features, let's take a look at some other existing terminology.

The Physical and Logical Drive

Most of you probably have at least a couple of physical disk drives in your system. In all cases, a *PHYSICAL DRIVE* is the actual hardware unit that you can touch, see, and feel. It may be a floppy drive of any kind or it may be a hard drive of ANY capacity — 20 MB, 30 MB, 80 MB or 380 MB. The capacity does not matter because it is a physical drive. In addition, physical drives of all kinds have a *PHYSICAL DRIVE NUMBER* that is usually set by a jumper or a set of switches on the drive. In many systems (excluding the AT compatibles), the physical drive number is usually set beginning with zero for each TYPE of physical drive in the system. To illustrate how this works, let's look at the physical drive setup that I have on my Z-100.

My Z-100 has two 5.25" drives, two 8" drives, and a single 26 MB hard disk. For the 5.25" drives, I have the physical drives identified as 0 (zero) and 1 by means of jumpers on the drives. Since the Z-100 has a separate connection for 8" drives on its floppy disk controller, these drives are also identified as physical drives 0 and 1. The Z-100 has a separate hard disk controller. And the jumper on my only hard disk is also identified as physical drive 0. When you run the PREP command to perform the low-level format on a hard drive, it asks you for the PHYSICAL drive number which is 0 for the first hard drive in the system. This remains true for PREP on any Zenith compatible system, such as the Z-386 — the first hard drive is still physical drive zero.

To continue with the example of the Z-100, the two 5.25" floppy drives (identified as physical drives 0 and 1) are identified in the system as LOGICAL drives A and B. The two 8" drives (also

identified as physical drives 0 and 1) are identified in the system as LOGICAL drives C and D. And the first partition on the hard drive (physical drive 0) is identified as LOGICAL drive E. As you can see, there generally is a direct relationship between the physical drive name (e.g., 0 or 1) and the LOGICAL drive name for each type of drive. In general, a LOGICAL name is nothing more than a special name that DOS uses to identify a physical device. These special logical names are reserved for exclusive use by DOS, and in the case of disk drives, single letters followed by a colon (e.g., A: or C: or E:) are used as logical disk drive names. For 5.25" floppy drives in this example, there is a direct relationship between the physical drive name (e.g., drive 0) and the logical device name (e.g., drive A:). For a hard drive, there is also a direct relationship between the drive number and the logical device name for the first partition. For the Z-100, the bootable partition is assigned a logical drive letter of E:. For a PC compatible, C: is the usual logical device name for the first partition on physical hard drive 0.

All of this is fairly straightforward for virtually all of the 8088-based systems, including Zenith and Zenith-compatible computers like IBM. All you need to do is set the jumper (or switch) for the correct of physical drive number for each drive type, plug in the cable(s), and you're ready to run. And when most people finally figured out how this convention worked, IBM decided to change it.

Unfortunately, IBM adopted a rather odd and twisted convention for connecting disk drives in the AT. If you have an 80286-based system, such as the IBM AT or Zenith Z-248 (or even the 80386-based Z-386), you may have noticed that both the floppy disk drive and hard drive cables to the controller contain a "twist" in a few of the wires. Although this twisted convention was apparently developed to eliminate the need to modify the jumpers on each physical drive from the factory setting, it has probably caused as many problems as it eliminated. The point of this discussion is that you do not need to use this twisted cable convention so long as you know how to set the physical drive jumpers for each drive. Now let's take a quick look at why there was a 32 MB partition size limit for older DOS versions.

The 32 MB Hard Drive Limit

If you are really interested in the details of why primary partitions were limited to 32 MB prior to MS-DOS 3.3 Plus or PC-DOS 4.0, you may want to read my October 1986 column where I explained the details of where that comes from, but it is essentially an arithmetic limit in DOS programming. Without getting into that again, suffice it to say that it was the limit for the partition size.

Zenith provided the capability to de-

fine and use up to four 32 MB primary partitions for MS-DOS as I mentioned earlier. It is especially interesting that Zenith was able to do this and maintain compatibility, despite the fact that PC-DOS still continued to have a limit of 32 MB that was accessible to PC-DOS on each *PHYSICAL* hard drive. Due to IBM's apparent reluctance to recognize that hard disk capacities were growing larger, many hard disk vendors were forced to develop some kind of "partitioning" software that would allow the entire capacities of these drives to be used by PC-DOS. To understand what happened here, it is important to understand how PC-DOS defined a DOS partition.

In PC-DOS, the FDISK command (similar to PART in function) would allow you to define one, and only one, partition that could be used by DOS. Although you could actually define more partitions (total of four) on any hard disk with these older PC-DOS versions (i.e., prior to 4.0) with FDISK, they were always called "Non-DOS" partitions meaning that they could not be accessed by PC-DOS and were typically used for other operating systems, such as Xenix. In this context, remember that these were called Primary Partitions and were defined in the Primary Partition Table. In particular, it is important to note that, in our current terminology, there could only be one Primary DOS Partition that was limited to 32 MB, regardless of the capacity of the hard drive. From what I can tell through my experimentation and research, PC-DOS version 4.0 still limits you to a single Primary DOS Partition, although its maximum size has been increased to 512 MB. From a Heath/Zenith user perspective, I should mention that my testing indicates that Zenith MS-DOS 3.3 Plus can still define up to four primary partitions, each of which also has the 512 MB size limit.

At this point, the question is: "If IBM still limits you to a single Primary DOS Partition, how can you add other partitions to a single hard drive that are also accessible by DOS?" Who wants to work with a single partition on a hard drive that has a 300 MB capacity? You would have to have lots of subdirectories, and you still have to cope with the maximum of 512 root directory entries which was not changed from earlier versions. The number of root directory entries is a critical number because you simply cannot exceed it. Sometimes users find that even 30 MB hard disks with a single partition are difficult to manage and maintain — consider the difficulty in managing and maintaining a 300 MB hard drive, not to mention how you would back it up!

The "solution" to this problem is found by creating a new type of partition, called an Extended Partition.

The Extended Partition

In the simplest terms, the Extended Partition is just a partition within a partition. I suppose it is called "extended" because it is specifically used to "extend" the capability of DOS to access hard disks larger than 512 MB. In addition, the use of the extended partition also allows PC-DOS to use other logical drive letters for different partitions on a single hard drive.

I mentioned earlier that the PRIMARY PARTITION TABLE is defined as a "list" (i.e., table) of up to four hard drive parameters, including the beginning and ending cylinder for each partition, that was initially created in the reserved area of a hard disk by a low-level format program, such as Zenith's PREP command; and a PRIMARY PARTITION is a partition whose entry is contained in the Primary Partition Table.

In contrast, the EXTENDED PARTITION TABLE is defined as a "list" (i.e., table) of up to four hard drive parameters, including the beginning and ending cylinder for each extended partition that is created by, and may be changed with, a DOS partitioning command, such as Zenith's PART or the IBM FDISK command. The Extended Partition Table is physically located at the beginning of EACH Extended Partition on the hard disk. The essence of this definition is that YOU can create an Extended Partition Table with the appropriate command, but you do not create a Primary Partition Table since that is done by a low-level format program. Similarly, an EXTENDED PARTITION is a partition whose entry is contained in an Extended Partition Table.

In theory at least, this explains what is generally happening in these new DOS versions, but there are some interesting side effects of this change, especially for Zenith MS-DOS users.

MS-DOS 3.3 Plus Side Effects

In order to remain completely compatible, there are a number of side effects which these changes have introduced. From a user perspective, the most significant change is that the Zenith ASGNPART command has been completely eliminated from this MS-DOS version because ALL partition assignments are now automatic. The option to change the Partition Assignment Flag from automatic to manual has, of course, also been eliminated from the DSKSETUP command because it no longer applies. And the Zenith PART command does not even remotely resemble its predecessor with the same name, but even that is not too surprising because of the major change in the hard drive partition size limit.

Although the new PART command is visually impressive because it displays in color and uses popup "windows", it is extremely difficult to learn to use when you are defining an Extended Partition because there is no easy way to tell exactly

what you are doing. Fortunately, the PART command usage is reasonably straightforward if you decide to create one or more Primary Partitions. The bad news is that the new PART command is quite difficult to understand and use when you are trying to create an Extended Partition. If you are looking in the manual for information on the new PART command, lots of luck because most of the information presented in this article is not documented in the manual in any form.

When you are creating an Extended Partition, the user interface for the new PART command needs some significant changes to make it easier to understand and use. The manual also needs some significant additions to explain what is going on, too. In the current manual, it looks like a few paragraphs have been added to the manual to "explain" the command, but the explanation is really bad which leads me to believe whoever wrote it did not understand the command or its uses.

In my opinion, both the new PART program and the command description need considerable changes to correct this problem, because the PART command is one of those really important, but rarely used commands. In all fairness to Zenith, I don't think the IBM FDISK command is much better. I hope that Zenith will develop a new version of the PART command that will be easier to use and explain in the manual. In general, I think that the PART command should be so easy to use that a user will not have to understand most of the theory that I have presented in this article — the command should be mostly self-explanatory to the point that the manual is not really required at all, except to document the theory of what is going on.

As most of you know, I generally have a very high opinion of the Zenith MS-DOS because I have found it is the best, most versatile, and most bug-free operating system available. That does not mean it is perfect by any stretch of the imagination, but it is still the best I have found, regardless of the difficulty in using the PART command for Extended Partitions. During the time I was working on this column, I read that IBM has introduced yet another fix for PC-DOS version 4.0 to correct some reported hard drive partitioning problems.

In any case, the new Zenith PART command was changed significantly, and even though it is not particularly user friendly or easy to use when creating an Extended Partition, it does do the job. Let's see how you might use it to define two partitions on a hard disk.

Using the New PART Command

The new PART command still displays partition numbers and types with their corresponding parameters, such as beginning and ending cylinders, and size.

You will see the partition type of "DOS (12)," indicating that the partition is less than 32 MB and uses 12-bit FAT entries, depending on the exact size of the partition. For larger partitions that are still less than 32 MB, you will see the usual "DOS(16)" partition type indicating that 16-bit FAT entries are used in this partition. The technical distinction between the FAT entry sizes is not too important for most users, although I have included it here so you will know there is a slight difference. You have probably seen at least one of these entries when you ran the old PART command to set up your hard disk. For MS-DOS 3.3 Plus, there are two new partition types.

For partitions larger than 32 MB, you will now see a "DOS(BIG)" partition type, and this partition also uses 16-bit FAT entries. And if you define an Extended Partition, you will see "DOS(EXT)" displayed as the partition type. All of that sounds fairly straightforward, and it does not really become confusing until you actually work with the PART command. In order to keep things relatively simple, I will use my 80 MB ST-4096 as an example and discuss how to create a 30 MB partition (drive C) for programs and a 50 MB partition (drive D) for data.

When the new PREP command finished, it defined the entire hard drive of some 80.5 MB (shown by PART) as a single partition with a partition type of DOS(BIG) as expected. Since PART does not have a "change" option (neither does FDISK), I had to delete that partition and then add a new Primary Partition of 400 cylinders (about 30 MB). The partition type shows as DOS(16) as expected, and I defined this partition as the bootable partition for drive C. At this point, I have changed the Primary Partition Table to show that 400 cylinders are used for this Primary Partition that is displayed as partition number 1. This is relatively easy since I selected the option Add a Primary MS-DOS partition (BOOTABLE) and followed the prompts with no problem. I still had 622 cylinders (about 50 MB) left that had not been allocated to any partition yet, and I want to allocate this space to drive D. This is the point where the use of the new PART command can become really confusing because it really is not clear what you have done until you finish doing it.

In order to allocate the remaining disk space, I selected the option to "Add an Extended or Logical MS-DOS partition (NON-BOOTABLE)" for the remaining 622 cylinders. Although the prompt sort of hints that this is a two-step process, I did not really figure it out until after some trial and error, mostly error. The first step is to define the Extended Partition as partition number 2, and it shows up as the DOS(EXT) partition type in the Primary Partition Table. In addition, the PART

command also creates an *EMPTY* Extended Partition Table for the remaining 622 cylinders, but I have not yet defined a "Logical MS-DOS partition" in that table — that's the second step.

Now that the Extended Partition Table is created, I had to go back through the prompts to add partition 1 to it (with the *PART* command, consisting of 622 cylinders to the Extended Partition Table. That's how, and when, you define a logical MS-DOS partition as the second step. I knew the second step was completed when I saw a "DOS(BIG)" entry as partition 1 in the Extended Partition Table that defined the 622 cylinders. The point is that you cannot access this extended partition for any reason, including *FORMAT*, until you also define it as a logical partition. In addition, there is nothing displayed on the screen that really gives you a clue as to what you must do, and are doing, until you finish it. If you are not extremely careful running this two-step process, you will find that you must run the *PART* command twice — once to define the Extended Partition in the Primary Partition Table (i.e., "DOS(EXT)") and create an empty Extended Partition Table, and once to define that partition as a logical drive as partition 1 (i.e., "DOS(BIG)") in the Extended Partition Table. When you have really completed defining an Extended Partition as discussed in this example, you will see different partition types, depending on which partition table you are looking at.

In this example, I have partition 1 defined as the first 400 cylinders as a Primary Partition with a "DOS(16)" partition type in the Primary Partition Table. Partition 2 is defined as a "DOS(EXT)" partition type in the Primary Partition Table for the remaining 622 cylinders (about 50 MB). When I go through the *PART* command displays and look at the partition information, *PART* asks if I want to display the Extended Partition information. By pressing a "Y" for Yes, *PART* displays the Extended Partition Table showing that Extended Partition 1 is defined as a "DOS(BIG)" partition with 622 cylinders. That's how I know that a logical partition has been defined. And it is important to know that a logical drive (drive D, in this example) is not really created and cannot be accessed until an entry is made in the Extended Partition Table.

Although I defined one Primary Partition that is bootable as drive C and one Extended Partition that is drive D, I did that primarily to ensure compatibility with all DOS programs that use Zenith MS-DOS 3.3 Plus. But I think one of the best features of Zenith MS-DOS has always been that it is so versatile, and this version is no exception.

Primary Versus Extended Partitions

I mentioned earlier that all partitions

in prior versions of Zenith MS-DOS were Primary Partitions which were defined by *PART* and could be accessed using the *ASGNPART* command. Although I decided to set up my hard disk with one primary and one extended partition to be consistent with the way that PC-DOS works, that is not a requirement in Zenith MS-DOS. It turns out that I could have decided to define both partitions as Primary, and my system would work the same way. That is probably a more straightforward use of *PART*, and it is consistent with the way that Zenith MS-DOS has worked in the past. In that case, the only real difference in version 3.3 Plus is that you do not need the *ASGNPART* command because all partitions are automatically assigned logical drive letters, assuming that you use the new *PART* command correctly, of course. If you want, you can use a combination of Primary and Extended Partitions to define logical drives up to drive letter Z. You can also define Extended Partitions within Extended Partitions, but that is so complicated that I won't try to explain it.

With PC-DOS version 4.0, you only have one way to allocate partitions for DOS — a single Primary Partition plus one or more Extended Partitions. With Zenith MS-DOS 3.3 Plus, you can use that convention, or you can define up to four primary partitions that can still be accessed by MS-DOS as in previous versions. Although I have never found any compatibility problems with the older Zenith MS-DOS versions that defined only Primary Partitions, I decided to follow the IBM convention to hopefully ensure maximum compatibility just in case. Perhaps the only valid reason to continue to use Primary Partitions is just so that you can boot a partition and if you are using multiple operating systems, that would be a good reason to continue that procedure.

Since the assignment of logical drive letters to each partition is now automatic in version 3.3 Plus, you may find it helpful to understand how logical drive letters are assigned.

Automatic Partition Assignment in 3.3 Plus

To ensure maximum compatibility, Zenith was forced to implement fully automatic assignment of drive letters to hard drive partitions, and how these drive letters are assigned is based on the PC-DOS convention. Let's consider what happened if you had two physical hard drives of, say 40 MB, on a system running PC-DOS 3.20. And remember that this PC-DOS version only allowed a maximum partition size of 32 MB for each physical hard drive that could be accessed by PC-DOS. In this example, the remaining 8 MB on each physical drive is assumed to be allocated for a Non-DOS partition.

When PC-DOS is booted in this ex-

ample, the DOS partition on the first hard drive (physical drive 0) is automatically assigned as drive letter C, and the DOS partition on the second hard drive (physical drive 1) is automatically assigned as drive letter D. Given the same situation, older Zenith MS-DOS versions would have assigned the drive letters in the same way which is the reason these versions were shipped with the partition assignment flag set to automatic. If you wanted to use more than one Primary Partition for these MS-DOS versions, you had to change the partition assignment flag to manual and use the *ASGNPART* command to manually identify additional partitions.

Zenith MS-DOS 3.3 Plus still follows that same basic rule. That is, the first Primary bootable partition, typically on physical drive 0, is assigned drive letter C; and the Primary DOS partition on physical drive 1 is assigned drive letter D. Then, the Extended Partitions on physical drive 0 are assigned drive letters, and Extended Partitions on physical drive 1 are assigned drive letters. PC-DOS 4.0 follows this same convention.

Once those drive letters are assigned, version 3.3 Plus can continue the assignment of drive letters to any "alternate" primary partitions that exist on either drive, but PC-DOS does not allow more than one primary DOS partition as I mentioned earlier.

As you can see, logical drive letter assignment flip-flops back and forth between the two hard drives. In general, logical drive letters are assigned first to the Primary DOS Partition(s), then to the Extended Partition(s) for MS-DOS and PC-DOS. If you have Zenith MS-DOS 3.3 Plus, it will also continue to assign logical drive letters to any "alternate" Primary DOS Partition(s) which are not permitted in PC-DOS.

Although the implementation of this hard drive enhancement is probably the most significant feature of version 3.3 Plus, there is one other enhancement that also should be mentioned. I still receive a lot of letters about using expanded memory on Zenith computers, and there seems to be a lot of confusion about expanded memory device drivers. This is related to the Zenith *EMM.SYS* expanded memory device driver that has been enhanced in the latest version.

Expanded Memory in 3.3 Plus

Version 3.3 Plus also includes an enhanced *EMM.SYS* that supports the EMS (Expanded Memory Specification) version 4.0 requirements. I have already received a letter from one Huggie who mentioned that he bought 3.3 Plus because the update card noted that a new *EMM.SYS* was included which supports EMS version 4.0. After he received the update, he was surprised and dismayed to learn that this EMS support only applies to Zenith mem-

ory boards which support EMS, such as the Z-515 memory board for the Z-386. Although that information has been documented in the Zenith manual ever since the EMM.SYS device driver was introduced, he apparently was not aware of the requirement. There still seems to be some confusion about expanded memory, so I will be specific.

Because of some minor differences in the way that the EMS can be implemented in hardware, virtually all manufacturers have developed an EMS driver for use with their memory boards that support it. More importantly, you can expect that one manufacturer's EMS driver will probably NOT work with another manufacturer's expanded memory board because of these minor hardware differences. That is, an EMS driver is custom-written for a specific manufacturer's expanded memory board design. If you want to update the driver for your specific brand of expanded memory, you MUST obtain a new version from the manufacturer of your memory board. Most manufacturers, including Zenith, are quite explicit about this point in their manuals, but that is based on the sometimes-bad assumption that everyone reads the manuals.

It is also important to note that Zenith has not developed expanded memory boards for all of the Heath and Zenith computer models, notably the 8088-based systems, such as the Z-150 series. The reason is that a number of third-party expanded memory boards work just fine in these older computers, and these boards include their own device drivers for the expanded memory. The point is that the EMM.SYS only works with Zenith

expanded memory boards, and since Zenith did not develop any for these older computers, you cannot use EMM.SYS with the non-Zenith boards in these systems. If you use any kind of expanded memory, just be aware that you will need to obtain an update for its EMS driver from the manufacturer of your specific brand of memory board. You may also need to check your application software manuals to see if you really need a later EMS version or not.

Powering Down

As you can see, the additional information to implement a large-capacity hard drive is somewhat difficult to explain and understand, especially given that the current manual is vague and generally unhelpful about this change. I have tried to supplement the manual by providing information about what is really going on, and I hope you find this article helpful if you are trying to implement MS-DOS 3.3 Plus on a large hard drive. For more specific information on exactly how I installed and set up my 80 MB hard disk, that article will appear in a future issue of REMark later this year.

For help in solving specific computer problems, be sure to include the exact model number of your system (from the back of the unit), the ROM version you are using (use CTRL-ALT-INS to find it), the DOS version you are using (including both version and BIOS numbers from the VER command), and a list of ALL hardware add-ons (including brand and model number) installed in your computer. The list of hardware add-ons should specifically include memory capacity (either added to an existing board or on any add-on

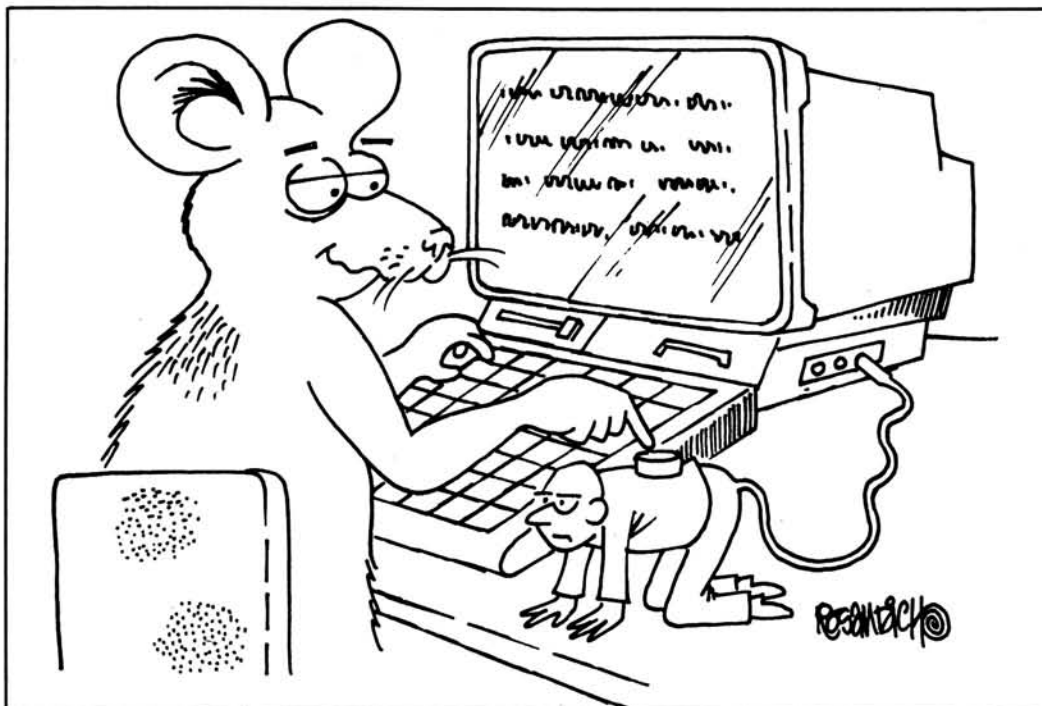
board), all other internal add-on boards (e.g., modems, bus mouse or video cards), the brand and model of the CRT monitor you have, and the brand and model of the printer, with the type of interface (i.e., serial or parallel) you are using. Also, be sure to include a listing of the contents of the AUTOEXEC.BAT 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 Heath/Zenith systems in general, be sure to include a self-addressed, stamped envelope (business size preferred) if you would like a personal reply to your question, suggestion, comment or request.

Products Discussed

MS-DOS 3.3 Plus	
(OS-51-3)	\$149.00 (List Price)
(mail order with update	
card only)	49.00
Heath/Zenith Computer Centers	
Heath Company Parts Department	
Hilltop Road	
St. Joseph, MI 49085	
(800) 253-7057	
(Heath Catalog orders only)	

Mace Gold Utilities	\$149.00
Paul Mace Software, Inc.	
400 Williamson Way	
Ashland, OR 97520	
(800) 523-0258	
(Orders only)	





HOME FINANCIAL MANAGEMENT

DONALD R. LEITCH
56 WEST VIKING DRIVE
CORDOVA, TN 38018

A REVIEW OF HFS-III FROM JAY GOLD SOFTWARE, INC.

"What are you going to do with your computer?" How many times have we all had to deal with this inevitable challenge from curious friends and family members? The answer was easy for me: I was determined to gain control of my home finances. To achieve this goal, I have chosen HFS-III, the Home Finance System from Jay Gold Software, Inc. Much more than a computerized check register, this package deals with the financial challenges of the average person — checking, savings and credit cards, but avoids the leap to a system loaded down with features most people don't need or a system that confuses its users with complex accounting terms and procedures.

Those who have been with Heath/Zenith long enough will remember the Jay Gold Home Finance System that ran under HDOS (What's HDOS??) and Micro-soft BASIC. It was quite an extensive program for its time. In fact, favorable reviews could be found in such respected journals as *REMark* and *InfoWorld*. The Home Finance System did a very creditable job of organizing and automating my personal finances and did it in much the same way I was already handling this chore — I didn't need a CPA to explain debits, credits, assets, and liabilities.

The program's major shortcoming was the two speeds at which it ran — slow and slower! Actually, this was a limitation imposed by the interpreted BASIC environment under which it operated. But nevertheless, as far as the user was concerned, a considerable delay (especially by today's standards) could be expected whenever a new module was loaded. The Home Finance System continued to evolve with the market, coming out with an improved CP/M version in 1984. This still was an interpreted BASIC program, but featured many assembly language

routines to enhance certain time intensive I/O functions and, yes, it was still slow when it came time to load one of the twenty-four sub-modules that formed the program. The CP/M version, of course, was a most welcome addition for those of us who had recently moved to the Z-100 computer. We could continue to manage our finances (an HDOS to CP/M data transfer utility was included) and enjoy the many features of this new computer.

The introduction of the industry standard microcomputer and Heath/Zenith's (along with other vendors') compatible systems led to HFS-III. It was two years in development, but one look should confirm that HFS-III was worth the wait — a 100% assembly language program that's fast, designed to anticipate problems before they occur and yet retains the flavor of its predecessors and runs on both the PC AND Z-100 computers.

The HFS-III package consists of three 5-1/4 inch floppy diskettes (or two 3-1/2 inch diskettes on request), a 106 page illustrated, typeset manual enclosed in a three-ring binder. The program requires a computer system with two floppy drives or a hard disk, 256K memory, monochrome or color/graphics video (supports color on PCs only), and any printer (most popular printers are directly supported).

Installation

Installation is accomplished with INSTALL.COM, a stand-alone program that completes the entire process. It begins with a check of its environment — H/Z-100 or PC. Then, using a screen display virtually identical to the main program, INSTALL checks the CONFIG.SYS file and modifies it if necessary, determines the desired drive/path configuration, and then prompts for disk changes, as needed, until you see the "HFS-III suc-



cessfully installed" message. Once installation is complete, you have a working system with demo data. Including the time to format four disks, floppy installation takes about eight minutes, and a hard disk system is ready to run in about three.

With Working Disk 1 in drive A or at the hard disk prompt, type "HFS" and the program displays its Startup Menu (See Figure 1). Screen displays remain consistent throughout the program. The top line displays the date, time, data path, and if active, status flags for Caps Lock, Insert, and Edit modes. The second line initially gives printer information, active function name, and version number. The bottom two lines show a legend for the various function keys which appear only when their respective functions are active. The Startup Menu, in addition to serving as gateway to the Home Finance Program, contains many useful default settings. These include password control, disk drive/path defaults, printer selection, user keys (handy macros for frequently used text strings), screen-saver, and display color (Red/Blue is my hands down choice).

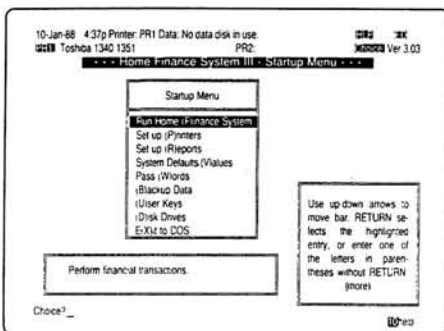


Figure 1

Menu choices are made by pressing the single key in parentheses in the menu item or by moving the highlight bar and pressing the RETURN key (the "point and shoot" method). If selected, help is provided in one window describing the currently highlighted menu choice and in a second whose information changes in context. Although they can be turned off, I found the help windows almost necessary to keep track of the many features available in this program. For example, the screen display only defines certain shifted function keys in a help window.

The (R)un Home Finance System option starts by asking for the location of your data disk, offering a default that was defined automatically during the installation procedure (it may be modified at any time). My system is on a hard disk, so it offers the default directory that I had specified at installation, C:\HFSDAT. (You are given the option of selecting the default or specifying a new drive and path so that you can keep unrelated financial records on separate paths or disks.) The program handles disk errors nicely with a message stating that the requested pro-

gram couldn't be found and options to try again or provide a new drive/path. Common errors like leaving the drive door open or using an incorrect disk are successfully trapped.

Setting Up Your Data

Once I had taken the time to familiarize myself with the program using demo data (this is *strongly* recommended in the manual), it was now time to create my own system. The Main Menu option (U)tilities allows you to create a new data file with or without previous data. The manual explains that once your system is set up, and you reach the end of the year (calendar or fiscal), you can carry forward all your fixed information — account names, expense codes, and recurring transaction information (called models), in addition to all ending balances. Using this process, the system will retain complete information for up to ten years!

In this case, however, I was creating a new system with no previous data, and so selected that option. I specified HFSDAT as my data directory and was warned that data already existed on that directory (it contained the demo data). When I responded to erase the data, it asked once more if I was sure; a touch of extra security that struck me as a bit too protective. After a few moments, a message confirmed that the new data files had successfully been created.

Next, using the Maintain (A)ccounts function, I added a checking account, a savings ("asset") account, and two credit accounts. I also entered my IRA account as a "parent" account, and the individual CDs in my IRA as additional asset accounts. The "parent" account keeps track of the balance in the whole IRA. When I enter deposits into the CDs to reflect interest earned, the "parent" account balance is automatically updated.

Reserve accounts are one of those features which characterizes HFS-III as the finance system "for the rest of us." Whether you call it "mad money," "overdraft protection," or some other similar term, many people like to "hide" some of their funds from themselves — money that's actually there as far as the bank is concerned, but not spendable. Reserve accounts do just that. Multiple reserve accounts can be created for any checking or asset account. The system allows free transfer to and from these reserve accounts and will maintain separate figures for the Total, Reserve, and Free (spendable) balances for each checking or asset account.

The next step in creating a new system is setting up codes. Select the (C)odes function at the Main Menu to establish the structure under which your expenses and deposits will be categorized. Although you may enter up to 100 expense codes and 15 deposit codes, I only

needed 24 for expenses and 5 for deposits. Some advance planning and analysis of your current sources of income and patterns of expense will make this process much easier. Two items are noteworthy in regard to HFS-III and codes: First, this program contains no budgeting feature, an omission that may not be too critical for many users and, if necessary, could certainly be handled external to the program. Second, careful planning of the code structure is worth the time and effort invested. Codes can be added or changed at any time, but only through this module. If a new code is necessary during transaction entry, you have to leave the transaction, go to (C)odes to enter the new code, and then return to the transaction. A minor delay, but nevertheless preventable.

Entering Transactions

Once the preliminary setup has been accomplished, you can go from the DOS prompt to entering transactions with a simple series of keystrokes, mostly carriage returns. One of the characteristics of HFS-III that becomes evident as you enter transactions is its consistent and predictable user interface and attention to technical details. The menu structure is hierarchical with a Lotus-like ability to "back out" using the ESC or F9 key. Whenever you're asked to provide an account name or code and aren't sure of your choices, press F4 and a window will appear with all the available choices that you can scroll through and even select by using the highlight bar to "point and shoot." All transaction entry screens are virtually identical to each other. Differences occur only as dictated by functional requirements. All fields on a transaction screen are accessible using the arrow keys and the Home/End keys. The contents of any field can be changed at any time before the transaction is actually saved. Even then, you can step back through the menus to (E)dit, identify your transaction by number or selection from a list (See Figure 2), and there it is again exactly as you entered it on an identical screen. Changes can then be made and the edited transaction re-entered into the record.

Check	Date	Payee	Amount	Code	T	Comment	R
M 891	02-Jan-88	Rocco's Cafe	12.33	41	0	Lunch with boss	N
M 892	05-Jan-88	Mammouth Discount	15.44	*	*		N
L 1008	02-Jan-88	Revcon Drugs	3.22	100	0	Aka Setzer	N
L 1009	02-Jan-88	Ms. Fooman's Fish	65.32	10	0		N
L 1010	05-Jan-88	Mammouth Discount	33.45	*	*		N
L 1011	10-Jan-88	Pretty Razy Bar a	11.93	41	0	Lunch w/ Joanne	N
06/20/88 BALANCE							
C 501	05-Jan-88	J. of Outback	40.00	50	T	1 year subscr	N
C 502	11-Jan-88	United American Te	30.00	50	T	1988 dues	N
C 503	11-Jan-88	Little Tots Day Ca	35.00	0	S	Sam Abbott	N
C 504	11-Jan-88	Fidelity Mortgage	100.00	*	*		N 505
C 505	11-Jan-88	PS Light and Gas	188.22	*	*		N 501
C 506	11-Jan-88	PlastiCard	600.00	*	*		N 503

Figure 2

For each checking account, the system will track three different series of

checks, i.e., the checkbook you carry, the one your spouse uses, and a third for the checks your computer writes. Once told which series of checks you're using, the system will offer checks in order, starting with the last used +1 but, if necessary, you can specify numbers out of order. In addition to date, payee and amount, you may enter a nineteen character comment.

HFS-III requires that all expenses and deposits be assigned at least one code to complete the transaction. For most transactions, there will only be one, but the system allows expenses and deposits to be split among up to fourteen different codes. As an example, my utility company itemizes electricity, gas, and water on the same bill. When I pay that bill, I use different codes to track each of those expense items separately. Each code entry consists of the code number or name (the program will convert the name to its corresponding number), the amount for that code, a tax flag for deductible items, and a separate comment tied specifically to that code entry. The real usefulness and flexibility (and potential confusion) of the code table becomes clear when you work with credit accounts.

Transfers Between Accounts

HFS-III handles transfers and credit accounts uniquely among home financial management packages. A transfer is described as the movement of funds between two accounts — checking to savings, checking to reserve, reserve to checking, etc. The screen for a transfer is similar to the single transaction screen, except that both sides of the transaction appear on the screen together and funds move between them in a single transaction. Credit accounts operate similarly in that the "paying" account always appears on the left of the screen and the "receiving" account on the right.

Credit accounts represent those institutions to whom we owe money, e.g., department stores, credit cards, banks, etc. Here, HFS-III once again deals with your finances in much the same way you do (See Figure 3). When the VISA bill arrives, you examine your purchases, verify that your last payment was entered and probably pay a portion of the total that's owed. HFS-III will, in addition to recording what's been paid, allow you to "pre-code" your expenses while you have the statement in front of you, pay all or part of what's owed, and save the information about what you still owe, including how you want these expenses coded when you actually make the payment.

All this flexibility in the code table can be a source of some confusion. Anytime a credit account with an unpaid balance is displayed, HFS-III will rebuild the code table with owed amounts just as they appeared after the most recent transaction. If a payment is made, HFS-III

17-Jan-88 8:11p Printer: PR1 Data: C:\HFSDAT
Yr:88(A) Free:11048960 Activity:57 Chkpm:17 Bal(Tot)3,924.52 ADD
100 Port Salut National - Checking 3302-499-0911

Code	Amount	T	Comment
2	68.88	0	

Amount: 68.88

15011 PS Light and Gas
Purch \$ 68.88 Fringh\$
Credit \$ PAID \$ 68.88
Current bal before txs 0.00
Current bal after txs 0.00

Date: 17-Jan-88

Arrears remaining \$ 0.00

Enter Backup Help

Figure 3

applies that payment to the first code; if that first code is satisfied, then the remaining payment is applied to the second code, and so on until all of the payment has been applied to a code. During this process, one of the codes may be partially paid, leaving an owed amount behind. This is a very logical process, and in fact, the program's designers may have been too logical since sometimes that's not the way we want our bills paid. This is a potential source of confusion, but once you understand the logic of the code table, you can use the function keys to adjust the physical order of the entries to assure that the correct expenses are paid first.

Reports

Any home finance management system, including HFS-III, is only as good as the information we enter into it. The discipline required to regularly enter data will be rewarded with a ready source of financial analyses, transaction records and professional looking printed checks.

Before HFS-III, the arrival of each bank statement signaled a continuation of a monthly battle of wits with my checkbook register. Invariably, a ten cent computational error would evade detection for hours (or, I could just take the bank's word for it??). Fortunately, HFS-III comes to the rescue. Any checking or asset account can be balanced, and once again, it's accomplished in much the same way most people are already doing it. First, enter the statement date and balance. Then, provide a list of returned transactions as reported by the bank (this is the only program I've seen that allows direct entry of checks by number, e.g., C101-125). HFS-III will then calculate the correct balance based on the information provided. Differences can be reconciled using the program's comprehensive "balance assist" features. An optional balance report lists the returned items, balance calculation, and a list of unreturned items — very handy in identifying that birthday check to your nephew which was never cashed.

For most Americans, April 15 is a time of dread and despair — an endless search for receipts, canceled checks, and bank statements either to prepare the return ourselves or to supply this information to

a tax professional. HFS-III can provide a report of tax deductible income and expense items (irrespective of their codes), a list of expenses by code or payee, or even a list of transactions that contain a specific comment. HFS-III helped me when a local department store advised that they hadn't received payment on my most recent statement. I asked HFS-III for a report of all checks to that payee and was able to quickly determine that not only had I paid the bill, but also that it had already cleared the bank!

Each month, I optimistically run a Financial Summary report to analyze my sources of income and expenses and determine if my net position is improving. And for those faced with the unfortunate situation of having many checks to write, or those who simply like the professional look of printed checks, HFS-III will print your checks using either a format supplied with the system or one that you design on screen with an easy-to-use editor.

Summary

While I have made every effort to remain objective in my description and analysis of this program, it should come as no surprise that I am very pleased with the features and performance of HFS-III. I have looked at many of the competing packages and find this program designed for the person whose financial needs encompass more than just a computerized checkbook, but don't include highly specialized activities in securities and speculative investments. Furthermore, I get the feeling that the developers at Jay Gold Software studied the methods and styles with which average people approach their finances and incorporated these concepts into a computer program. You are not forced to adapt to a set of complex, unfamiliar procedures. Managing home finance is at the top of my list of computer applications and HFS-III has certainly met this need.

Acknowledgements

Thanks to Jay H. Gold, M.D., who provided historical insight into the development of his product. Screen photos courtesy of Jay Gold Software, Inc.

HFS-III Home Finance System \$99.00
Jay Gold Software
P.O. Box 2024
Des Moines, IA 50310
(800) 541-0173



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

*** Z-100 SERIES SOFTWARE ***

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

*** IBM COMPATIBLE SOFTWARE ***

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

*** ZENITH LAPTOP COMPUTERS ***

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

*** VIDEO MONITORS ***

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

*** ZENITH PC COMPUTER UPGRADES ***

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

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

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

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

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

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

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

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

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

*** Z-100 SERIES COMPUTER UPGRADES ***

High Density 1.2 Meg Drives. External floppy drive set-up complete with drive, power supply, case and cable. Ready to connect to your 8" floppy controller
Single Drive Unit 234.00 Dual Drive Unit \$351.00
Bare Drive and Cable for internal mount \$136.00

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

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

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

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

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

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

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

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

UCI EasyWin Winchester Drive Systems at reasonable prices. Complete Hard Disk Systems for mounting inside your Z-100. Systems complete with Seagate Drives, 21 MEG \$598.00, 31 MEG \$634. System without Drive \$317.00

CDR Z-100 Speed Module Run your Z-100 Computer at 7.5 MHz. Installs easily with no soldering. Externally switchable between Speed and Normal mode. Payload \$44.00

*** FLOPPY DISK DRIVES ***

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

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

*** SEAGATE HARD DISK DRIVES ***

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

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

* PAYLOAD CUSTOM ASSEMBLED COMPUTERS *

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

Example #1

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

Example #2

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

Example #3

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

MS-DOS 3.3 Operating System \$85.00

PAYLOAD COMPUTER SERVICES



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



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

POWERING UP

William M. Adney

P.O. Box 531655

Grand Prairie, TX 75053-1655

Copyright © 1988 by William M. Adney. All rights reserved.

Selecting and Setting Up a Hard Disk

One of the best ways to improve the performance of your system is to add a hard disk. It is the most cost-effective way to improve overall system performance even in systems that run at the 4.77MHz clock speed. The speed of disk input/output (I/O) is one of the major factors that can influence the apparent speed of your system. On a floppy disk that typically rotates at 300 RPM, the disk I/O speed is dependent on the mechanical drive characteristics and its rotation speed. If you know that a hard disk rotates at 12 times that (3,600 RPM), it becomes obvious that it will nearly always improve the apparent processing in your system. If you have any programs that use a special file type called an overlay (e.g. OVR or OVL file type), all of these programs will appear to run faster because they can be read from the disk at much higher speeds. You may not see nearly as much improvement in some memory-intensive programs, like spreadsheets, except when you read or write files to disk.

The purpose of this article is to help you choose a hard disk, and once you get it, help you set it up so that it can be used most effectively on your system. In order to understand what the implications of a hard disk are, it is best to talk about a hard disk system.

The Hard Disk System

A hard disk system usually consists of two components: a plug-in board that fits inside your computer and the hard disk it-

self. The plug-in board is called a hard disk controller, and it acts as the physical link or interface between your computer and the hard disk. Your computer already has a floppy disk controller that performs a similar function for the floppy disk drives. Most of the current Zenith desktop computers, except for the Z-150 series and some of the Z-200 series, have a "dual" disk controller card that can be used with floppies and hard disks. For those systems which do not have a dual controller card, you will need to buy a hard disk controller card in addition to the hard disk itself.

A typical hard disk on a PC compatible computer has 17 sectors per track and spins at 3,600 RPM. Without getting into a lot of the technical details, these hard disks use a technique, called MFM or Modified Frequency Modulation, to encode data on the disk. Today, this is still the most common type of hard disk, but there is another encoding method that is becoming more popular. This newer encoding method is called Run Length Limited, or RLL, and there are two common RLL modes that you may find.

As of the time of this writing, the term "RLL" technically refers to "(2,7) RLL" that is also called ERLL (Enhanced RLL). It has a 50% data density increase over standard MFM encoding and usually has 24 sectors per track. The newest RLL encoding technique, called "(3,9) RLL" or Advanced RLL (ARLL), has a 100% increase over standard MFM encoding

and usually has 31 sectors per track. ARLL is not too popular yet because of its technical complexity and associated high cost. Because more data can be written on each track (a sector still contains 512 bytes), using the RLL technique effectively increases hard disk capacity.

When you select a hard disk system, it is important to choose a matched controller and hard disk. Although this is becoming less of a problem today, a "standard" MFM hard disk generally does not work well with an RLL controller. The reason is that the usual MFM hard disk is only tested and certified for 17 sectors per track, and when the data density is increased for RLL recording, an MFM certified hard disk probably will not be reliable. The moral is: If you are looking at RLL controllers, be sure that you also get a hard disk that is certified for the appropriate type of RLL operation.

Before we get too much into the details of getting a hard disk and controller, there are two specific features that your computer has which are related to hard disk systems.

Zenith System Features

Heath and Zenith computer systems have some features that are not available on other brands of computers. One of the nicest features is that the user can access the ROM Monitor to perform various tasks and control the system. You can access the ROM Monitor (and see what version you have) by using the CTRL-ALT-

INS key sequence. To see all of the various commands that you can use, type a question mark (?) and press RETURN. This displays a help screen showing all of the commands. Although most of the commands are intended for technical uses, you might find the TEST command useful because it can help you test your keyboard and disk drives. But the Zenith ROM has one special feature that can be very useful for all computer users, especially if you have a hard disk. By using the appropriate command, you can boot your system from ANY disk drive, either floppy or hard disk. This can be especially useful if you have a problem with your normal boot drive that is activated with a CTRL-ALT-DEL. It is easy to do.

The basic boot command is a "B" followed by the drive type -- either "F" for floppy or "W" for winchester (hard disk)-- followed by the drive number "0" or "1". A summary of these commands for drives A through D is shown in Figure 1.

Drive	Boot Command
----	-----
A	BF0
B	BF1
C	BW0
D	BW1

Figure 1
Boot Commands for Zenith ROM

The capability to selectively boot from any disk drive is one of the unique features of Heath and Zenith computers. There is one other feature that is particularly noteworthy. To understand the benefit of this feature, let's take a look at how a typical IBM computer boots a hard disk system.

A typical IBM computer, and many clones, ALWAYS attempts to boot the system from floppy drive A. When one boots any of these computers with the CTRL-ALT-DEL key sequence, the red light for floppy drive A comes on, and the system will sit and "grind" for a few seconds in an apparent attempt to read a bootable disk in the drive. Finding none, the system will then attempt to boot the system from a hard disk which will hopefully be successful. If you have a non-bootable disk in drive A, and try to reboot one of these systems, you will very cleverly be told that you have a "non-system disk" in the drive. Then you have to open the door on drive A and reboot the system again. Depending on the specific brand of computer and hard disk controller, this process can take as long as 20 seconds.

If your computer has a Zenith hard disk controller, this intermediate step is bypassed, and the system is able to boot directly from the hard disk without the intermediate step of checking the floppy

drive. For the Z-150 series computers, you can set the default boot drive by means of a DIP switch inside the computer. For the 80286 and 80386 systems, the default boot drive is defined by changing a parameter with the SETUP command in the monitor ROM. In either case, the boot process is much faster in Heath and Zenith computers because the intermediate step is eliminated.

Many of the older 8088-based computers also had the capability to change clock speed by using a switch on the back of the unit. This allowed you to use the standard IBM 4.77MHz clock speed as well as a faster clock speed of 8MHz. The capability to use the 4.77MHz clock speed can be important because some forms of copy protection are clock-speed sensitive.

These three features may influence your decision as to what kind of a hard disk system you need because there is one other related issue that is important to understand. For lack of a better term, we'll call it compatibility.

Hard Disk System Compatibility

The discussion of hard disk system compatibility tends to become an emotionally-charged issue because you will probably be able to find someone who has had a lot of difficulty either in the installation or setup. In most cases, allegations of incompatibility of Zenith systems with various "PC compatible" hard disks and controllers have been caused by a lack of understanding of the hardware design and requirements. This, coupled with the rather high prices of the Zenith hard disk upgrades, has conspired to lead to a certain amount of frustration. To help you make an informed decision on what kind of hard disk to buy, let's take a look at each side.

All hard disk controllers have a ROM that is used to provide various control features for a hard disk. In particular, Zenith hard disk controllers have a unique Zenith ROM. It is the Zenith hard disk controller ROM, in conjunction with the system ROM, that gives you the capability to directly boot a hard disk drive using the commands shown in Figure 1. In addition, a Zenith controller ROM also checks the status of a DIP switch (e.g. on the Z-150 series) or the settings defined by the SETUP program (on 80286 and 80386 systems) so that a hard disk will boot directly with no intermediate step of checking drive A. The Zenith controller ROM is also designed to work at all clock speeds in all Zenith computers.

Although you can be assured that the hard disk upgrades you see in the Heathkit catalog will work with the listed systems, the bad news is that they are expensive. In general, they are more expensive than most other hard disk upgrades you will find, but the good news is that

they have all of the standard Zenith features and capabilities. Still, most of us are concerned about price, and there is another alternative.

If you are considering a hard disk, I recommend that you take a look at the REMark advertisements by various vendors. You will generally find they offer very competitive prices with the added advantage that they understand Zenith computers and can provide you with technical support if you have any problems. If you have no experience with hard disks, it is particularly important to buy one where you can get some help if you have problems. And if you buy a non-Zenith hard disk system, be absolutely sure that you obtain a listing of its characteristics so that you can answer the technical questions shown in Figure 2.

1. What is the drive Type (80286 or 80386 systems only)?
2. Does the hard disk require servo track information?
3. How many cylinders does it have (decimal number)?
4. How many heads does it have (decimal number)?
5. What is the starting write precompensation cylinder (if any)?
6. What is the landing or shipping zone cylinder?
7. What is the starting reduced write current cylinder (if any)?

Figure 2
Hard Disk Characteristics

We will look at how the answers to these questions may affect the setup and usage of a hard disk later, but it is a good idea to ask the dealer for this information just in case. In many cases, it is quite unlikely that the dealer will have this information, although you will probably find that you must start with the dealer in order to get the manufacturer's address. If you decide to go that far, I also recommend that you just get the technical manual from the manufacturer for your specific hard disk, but be aware that some manufacturers charge as much as \$50 for that manual. I have found that a manufacturer's manual and specifications for a hard disk are a worthwhile investment.

Before we get too far away from a discussion of hard disks and compatibility issues in general, it is appropriate to take a brief look at another kind of hard disk system

Hard Disks on a Card

You can find any number of hard disks that are combined with a controller card, such as the Hard-Card. The biggest advantage is that these hard disks are very easy to install: just remove the cover from your computer, plug it in a slot, replace the cover, and perform some setup commands. The biggest disadvantage of this

approach is that it is more expensive and much less flexible than a separate controller and hard disk. And if you have an 80286 or 80386 system that already has a DUAL controller card (i.e. floppy and hard disk), you may find that a hard disk on a card does not work very well or at all. Let's examine each one of these disadvantages in more detail.

Because of the technology involved, you will pay a premium of about \$50-\$100 for a hard disk on a card compared to the separate units. And if you ever decide that you need a hard disk with a larger capacity, it is not just a matter of buying another hard disk; you will need to buy ANOTHER controller as well which costs even more. You will also find that some of these units effectively require TWO slots in your system because of the thickness of the hard disk itself. If you have a lot of other boards in your system, this can also present a cooling problem because the hard disk card unit may, of necessity, need to be located next to a memory board that generates a lot of heat.

The flexibility issue also has another aspect to be considered: repair. When the hard disk OR the controller fails, you will most likely have to return the entire unit to the manufacturer for repair. In that situation, you do not have the option of replacing or repairing only the failed unit — the disk drive or the controller — because they are together. Although I don't know for sure, I would guess that it would be far more expensive to repair a hard disk on a card than it would be for separate units.

If you have an 80286 or 80386 system that has a DUAL disk controller card, do NOT even consider buying a hard card for the very good reason that it may not work very well, if at all. Without going into all of the technical details, suffice it to say that there may be a conflict between the two controller cards because one is "connected" and one is not. The best you can probably hope for in this situation is that the hard disk on a card will simply be slow because the system is "confused" by the two controllers and trying to resolve the conflict. Sometimes the conflict may be successfully resolved and sometimes it may not. That will lead to erratic and unreliable hard disk operation. In the worst case, the hard disk on a card may simply not work at all. For those of you who are technically inclined, you may want to explore some of the mysteries of something called an Interrupt Request (IRQs) to see how this kind of conflict could happen and why the system may not be able to successfully resolve it.

It is also ridiculous to spend the extra money for a second controller if you already have a dual controller card in your system aside from the fact that it may not work.

As you can see, I do not recommend

any of the hard disks on a card for a number of reasons. They are expensive, inflexible, and simply may not work in some systems.

That covers the general background of things you should know about hard disk systems, but now let's take a look at how to choose the best combination of capacity and "speed" to fit your system.

Capacity Versus Speed

A hard disk is rated in terms of capacity for data storage (in megabytes) and "speed." In general, you will probably want to buy the largest capacity hard disk you can afford, although a 20-30MB unit is quite adequate for most users. The speed of a hard disk is related to how fast it can move the read/write heads to access data on the drive. This speed is given in milliseconds (abbreviated ms.), and common speed ranges include about 65ms. for a "slow" drive to down around 28ms. or so for a "fast" drive. As you might expect, the cost of a hard disk is based on both its capacity and speed.

For most people, capacity is more important than speed. Speed only becomes important when you are routinely working with a lot of files or very large files such as in a data base application. Also, you can live with a slower hard disk, but you can't change its capacity; hence, the reason I suggest buying the largest capacity hard disk you can afford.

Installing a Hard Disk

Before we get into some of the details of installing a hard disk, I thought it might be interesting to show you a photo of what a hard disk really is. Take a look at Photo 1.



Photo 1
A Hard Disk

On the left side, you will see the "platters" that store data. As you move to the right, the Read/Write heads are attached to a single movable arm assembly that has its pivot point on the lower part of the photo. In the upper right-hand corner is a pulley-like arrangement which is connected to a stepper motor that moves the Read/Write heads to the appropriate cylinder (or track) on the media surface. This particular drive is a Shugart that has two platters and four Read/Write heads.

By the way, do NOT ever disassemble your hard disk to see what is inside unless you intend to throw it away. Hard disks must be disassembled and repaired in a "clean room", and you may contaminate or ruin it beyond any hope of repair. The drive in this photo was already ruined, and it was loaned to me for this article by the Dallas Heathkit store. Now let's move on to the installation process.

There are so many different models of Heath and Zenith computers that I will not go into a discussion of how to perform the physical installation of a hard disk. The hard disk upgrades listed in the Heathkit catalog include instructions on how to perform the installation on each computer system. If you decide to buy a non-Heath/Zenith hard disk upgrade, you might want to ask a friend who has done it before to help you. Even though the installation is not particularly difficult, everything has to be EXACTLY right or the hard disk will not work. Assuming that your hard disk is installed, let's move on to the setup of the system.

The General Approach

Before we get into all of the details of the setup process, I would like to take a minute to look at the overall procedure used to prepare a hard disk for use.

If you have an 80286 or 80386 computer, you will use the ROM Monitor's SETUP program to define the drive Type. Except for that, the remaining steps are the same for all other systems.

The first step is to run the PREP command to test and prepare the hard disk. Then you will run the PART command to identify that one partition is bootable so you can boot your system from it. The next step is to use the FORMAT command with the /S switch to copy the system files to the partition so it is bootable. And finally, the last step is to copy all software to the hard disk.

As we go through each step, I will explain what is going on and why the step is necessary. Although this procedure is generally documented in your Zenith owner's manual, there is not too much discussion of why the steps are performed in the order listed. As we go through each step, I will describe what is happening and why it is important.

Although the general setup procedure contains the same steps, some of the details may be slightly different depending on which MS-DOS version you are using. For that reason, all details about the various commands are specifically intended to include features available in Zenith MS-DOS versions 3.20 and 3.21. It is reasonable to expect that later versions will have about the same features and capabilities.

The SETUP Program

One of the features of PC compatible

computers is that they were intended to make the setup of a hard disk fairly easy. Because the system must know the characteristics of an installed hard disk (i.e. see Figure 2), the first IBM hard disk computer, the XT, contained the characteristics of about 16 different hard disks in the system ROM.

Because of compatibility considerations, the Zenith system ROM also contains the characteristics of different hard disks, although the list is NOT totally identical to IBM's list. Different models of Heath and Zenith computers have different ROMs which provide support for different hard disks. For example, you can install a full-height hard disk in a Z-158 or Z-159 system (by sacrificing one of the floppy disk drives — not recommended), but the Z-148 simply does not have enough physical room for a full-height hard disk — you must install a half-height hard disk in a Z-148. Although most hard disks are pretty much in accordance with the IBM "standard" up to 32MB, you might find a special deal on one that is not. Because of these possible differences, I still recommend that you buy a hard disk and controller (if needed) from a vendor who understands Zenith computers. If there is any question about this, be sure to get the answers to the questions listed in Figure 2 because there is a way to install just about any kind of hard disk on a Zenith system as long as you have a current version of Zenith MS-DOS.

If you have an 80286 or 80386 Zenith desktop system (not a laptop), the first thing you must do is run the ROM Monitor's SETUP program to define the TYPE of hard disk you are using. All you do is use the right cursor key to move to the Drive 0 (the first hard disk) area shown on the menu, and you can adjust the drive type by pressing the SPACE BAR to change it. Even if you don't know the exact drive type number (e.g. 1 through 40 or so), you can still select the correct drive type if you know the answers to all the other questions shown in Figure 2. Note that all parameters should be an EXACT match to the questions for your drive.

For example, one of my '248 hard disks is a Type 2 that SETUP tells me has 615 cylinders (numbered 0 through 614), a Ship Zone in cylinder 615, a "Precomp" of 300, 4 heads, 17 sectors, and a formatted capacity of 21MB. If you do not find an exact match for the characteristics of your specific hard disk, you will want to set the drive type to 1 and override the ROM characteristics with the PREP command parameters that will be discussed later in this article. In addition, you should refer to the Zenith MS-DOS manual's appendix that provides additional information on "Extended Hard Disk Support" for version 3.2 and later. For any other questions about the SETUP program, refer

to your owner's manual.

At this point, I will assume that you are implementing a single hard disk on a Zenith computer with a Zenith hard disk controller using Zenith MS-DOS 3.20 or later. If you have any non-Zenith hardware, you can follow the steps in the order shown here, but you may have to make adjustments in some of the details. Now let's begin preparing the hard disk.

PREP the Hard Disk

At this point, you should be ready to run the PREP command. Power-on your computer system, and be sure to run the ROM Monitor's SETUP first if you have an 80286 or 80386 desktop system. If you do not have an exact duplicate of your Zenith MS-DOS distribution disks, take a minute to use DISKCOPY to create an exact backup copy of each one. Then insert the exact backup copy of disk 1 into drive A, and reboot the system. Because this is an exact duplicate of the distribution disk, the AUTOEXEC.BAT file will begin execution of the SETUP.EXE program which will ask if you want to continue with the SETUP. At this point, enter an N so you will return to the DOS command prompt. Note that this is NOT the same SETUP command that is available in the 80286 and 80386 system ROMs. I should also mention that it is absolutely essential to locate the proper disk and run all of the commands from your backup copy of the distribution disk in drive A. More importantly, it is critical that the commands be performed in the order listed or you will end up with all kinds of strange problems and/or error messages.

If you know that your hard disk's characteristics are established in the system ROM, all you need to do is verify that the PREP command is on the current drive. PREP will ask if you are sure you want to proceed (type a Y), and it will ask for the drive number (type a 0 — zero). In most cases, you will not be installing a cartridge drive (usually type an N), and the hard disk will not normally require servo information (type an N). PREP will then begin initializing and testing the hard disk. This process will take from 40 minutes to several hours to complete depending on your system and the hard disk's capacity and speed.

The basic function of PREP is to perform a multi-pass media check (6 passes on a PC compatible) to identify bad sectors or areas and initialize the hard disk. PREP performs a special kind of initialization, similar to FORMAT, on a hard disk which is called low-level formatting.

The LOW-LEVEL format is required to add the special "stuff" to a hard disk that is not necessary on a floppy. For most hard disks, this special stuff is essentially a partition table and some other things which are added to a hard disk in the form

of a Boot Record. The Boot Record is actually created by the low-level format program (e.g. PREP) on the hard disk, but you can change some of it with the Zenith PART and DSKSETUP commands.

Before PREP begins the low-level formatting, it checks the ROM for hard disk characteristics. For an 8088-based system, it finds the exact or closest MATCH for that specific hard disk. If it does not find an exact match, it attempts to find the closest characteristic that IS in ROM. For that reason, you may find that the hard disk does not have the capacity you expect when you finish with the FORMAT command. If you find that situation, you will need to rerun PREP using the /Q (Query for characteristics) switch. When you use the /Q switch, you MUST know the answers to the questions listed in Figure 2. By the way, I will mention again that you MUST have Zenith MS-DOS version 3.20 or later to have this feature.

When PREP has completed the low-level formatting, it is time to run the PART command.

PART the Hard Disk

The PART command performs two major functions: it defines which partition is bootable (even if there is only one), and it can divide a hard disk into multiple partitions. In this context, a PARTITION is a specified "piece" of the hard disk that can be assigned a drive letter. Since we have assumed you have a drive with a capacity of less than 32MB, it is usually best to NOT subdivide this into multiple partitions. If you have a drive with a capacity of more than 32MB, I recommend that you define partitions of slightly less than 32MB each to ensure compatibility with all software.

The primary reason for running PART is to define which partition is bootable which allows you to boot the system from the hard disk. Assuming that you have a hard disk with less than 32MB, PART will ask you if you want to continue (type a Y), and then you will see a "menu" showing the F1 is used to partition Winchester drive 1. Press F1, and you will normally see a display showing the Winchester unit number (normally 0 for the first drive), the Default boot partition (shows as "None" at this point), and the Maximum cylinder number for that hard disk. In most cases, you will also find that the first partition has been allocated as a "DOS" partition so that all available cylinders are used. An information message shows that you can press the F1 key (do it) to Select the default boot partition, and it will ask you to select a partition number, so you press a 1 (for the first partition). Note that the display now indicates that the "Default boot partition =1." Now exit the program, reboot the system, and you are ready to FORMAT the drive.

FORMAT the Hard Disk

A hard disk must be formatted with the `FORMAT` command just like a floppy disk. For a hard disk, this is called `HIGH-LEVEL` formatting to differentiate it from the low-level formatting performed by `PREP`. You will need to use the `/S` (System) switch to make the partition bootable that was specified with the `PART` command (use the "`FORMAT C:/S`" command).

At this point, your hard disk is completely set up and ready for use. You have performed the low-level formatting with `PREP`. Then you ran the `PART` command to specify the bootable partition. And then the high-level formatting was done with the "`FORMAT C:/S`" command to transfer the system files to the hard disk so that it was bootable. You can use the "`DIR C:`" to be sure that `COMMAND.COM` is on the hard disk, and you may want to double check by running the "`CHKDSK C:`" command too. If you made any changes with the Zenith `CONFIGUR` program, you should rerun `CONFIGUR` to make those same changes or use the `SYS` command from a floppy that has the configured system files. If you have done everything correctly, you should find that `CHKDSK` reports "2 hidden files" which are the BIOS and the System Kernel. Now for the real test.

Boot the Hard Disk

If you do not have a Zenith hard disk controller, open the door to drive A, and use the `CTRL-ALT-DEL` key sequence to reboot the system. If you are using a Zenith hard disk controller, use `CTRL-ALT-INS` to access the monitor ROM. Type `BW0` (followed by `RETURN`) to boot the hard disk, and the system should boot normally. If it does not, rerun the `PART` command to be sure that you have specified partition 1 as the default boot partition. And be sure that you ran `FORMAT` with the `/S` switch so that the hard disk is bootable.

Loading System Software

The next step is to be sure your hard disk is set up the way you want it. The easy way to start is to create an `AUTOEXEC.BAT` and a `CONFIG.SYS` file in the root directory. Since you don't have an editor handy on the hard disk yet, you can use the "`COPY CON`" trick that I discussed in a previous article. A basic `AUTOEXEC.BAT` file is shown below:

```
DATE
TIME
PATH C:\;C:\DOS;C:\BATCH
PROMPT $P $Q$Q$
```

If your system has a real-time hardware clock, such as an 80286 or 80386 system has, you will not need the `DATE` and `TIME` commands, and they should be omitted.

Then you can add a basic `CONFIG.SYS` file that includes the following:

```
BUFFERS=30
FILES=25
```

These are examples only, and if you have any questions about this, be sure to reread the article on "Batch Files and `CONFIG.SYS`."

Now create some subdirectories using the following commands:

```
C:\>==>MD \DOS
C:\>==>MD \BATCH
C:\>==>MD \CONFIG
```

The `\BATCH` subdirectory should contain all of your batch files, and I suggest that you immediately `COPY` the `AUTOEXEC.BAT` file to that subdirectory so that you will have a backup in case you happen to erase all files in the root directory.

The `\CONFIG` subdirectory should contain a backup copy of `CONFIG.SYS` plus any other variations that you use. Again, this provides a backup copy in case the root directory gets erased.

Change to the `\DOS` subdirectory and copy all files from each of your backup distribution disks by using the "`COPY A:.*`" command. After that is completed, use the "`DIR/W`" command to review all files, and carefully delete the ones you do not need. In general, this includes: `SHARE`, `SELECT`, and all of the `KEYBxxxx` programs. I also suggest that you delete `PREP` since it is bad practice to keep a program on a hard disk that can completely and irrevocably destroy everything on the disk. I also recommend that you delete the `RECOVER` command because it can screw up your hard disk beyond any hope of recovery unless you **REALLY** know what you are doing.

Now that all of the basic system files and commands are loaded, it is a good idea to reboot the system to be sure everything is set up correctly. Now you can load your application software.

Loading Application Software

Plan to spend a considerable amount of time to load all of your application software on your hard disk. In most cases, you will have to find your distribution disks and begin the installation process again. Most of today's application software requires a knowledge of the disk setup — that is, you must install an application in a subdirectory, and you must define that subdirectory name during the installation. You cannot usually just use the `COPY` command to copy major applications to a hard disk because of this requirement. Depending on how many applications you have, the installation process can take the better part of a weekend.

As I have mentioned before, keep your subdirectory names short and descriptive. Remember, there is no reason

that a subdirectory cannot have the same name as a program. If you are installing Microsoft Word, Word Perfect or WordStar, use subdirectory names like `\WORD`, `\WP` or `\WS`.

When You Are Loaded

After writing that section heading, I almost decided to change it because I really meant "When your hard disk is loaded." A little humor (very little!) helps in an article like this, and sometimes a grin is real helpful when you are working with a computer. In any case, you have now spent a considerable amount of time setting up your hard disk and installing applications, and it's time to develop a habit that probably will, at some point, save your sanity, or at least most of it.

Now that you have a few megabytes of programs, and perhaps data, on your hard disk, what would you do if you made a mistake and erased a critical subdirectory or had a hard disk failure? It is never too soon to backup a hard disk, and you have several options.

Back It Up

In general, you have two options for hard disk backup: software and hardware. The importance of a hard disk backup has even been recognized in DOS because it includes the `BACKUP` and `RESTORE` commands. The general syntax of the `BACKUP` command is shown in Figure 3.

```
BACKUP [hard-d:][\path]afn floppy-d:[/p]
```

Figure 3
BACKUP Command Syntax

To backup all files in all subdirectories, you will need a bunch of floppy disks (about 3-360K floppies for each MB on the hard disk), and you can use the following command:

```
C:\>==>BACKUP C:.* A:/S
```

This example shows that you are backing up all files (`.*`) from drive C to floppy drive A, and the `/S` switch means to include all subdirectories in the backup.

To `RESTORE` files to the hard disk, you can use the general syntax shown in Figure 4.

```
RESTORE floppy-d:[\path]afn [hard-d:][/p]
```

Figure 4
RESTORE Command Syntax

You can use the `RESTORE` command to restore all files in all subdirectories to the hard disk using the following:

```
C:\>==>RESTORE A:.* C:/S
```

Or, you can restore all files in a single subdirectory with:

```
C:\>==>RESTORE A:\MYDIR\.* C:
```

Or, you can restore a single file using a syntax like:

C:\>==>RESTORE A:\MYDIR\MYFILE.DAT C:

The BACKUP and RESTORE commands both have additional forms and switches, and you should refer to your DOS manual for additional information. Although I have included information about using the DOS BACKUP and RESTORE commands, I do not use them for two reasons. First, they were unreliable in at least one DOS version (I think it was 3.10), and second, they are very SLOW.

Because of the problems with these DOS commands, there is a lot of backup software on the market today. My favorite is DSBACKUP+ which is relatively inexpensive, but it is reliable and fast. It will backup about a megabyte a minute, and how long it actually takes you to perform a backup is basically limited by how quickly you can change floppy disks.

Unfortunately, most users' backup strategy is not to do it because it takes too much time; and in order to be effective, you must backup your hard disk on a regular basis, say every week or month. Another alternative is to buy hardware, usually some kind of tape backup system, that allows you to take regular backups without a lot of user intervention such as changing floppy disks. Tape backup is an excellent way to backup a hard disk because you don't have to be there during the process — you can run a backup during lunch or a coffee break.

Although tape backup systems are nice, most of them have a singular disadvantage — they are relatively expensive to the tune of \$500 and up (mostly up). Even if you find a good deal on a tape drive, the cost of the special tapes that must be used with most of these units is on the order of \$20 each or so. It is not too difficult to spend \$700 on an internal backup unit with the cost of tapes and such. And if you need an external unit with a power supply, you need to plan on getting very little change back from \$1,000 by the time you buy tapes and everything. But if you already have a standard Video Cassette Recorder (either VHS or beta), you can save a lot of money with the Imager.

The Imager system consists of a full-size board that fits in an 8-bit slot on a PC compatible computer, a disk of software, connecting cables for the VCR, and a manual. Because the Imager fits in a standard 8-bit slot, it works with nearly all Zenith PC compatible computers (except laptops of course) up to and including the Z-386. The only real requirement is that you must have space with an 8-bit slot for a full-size board.

The Imager is remarkably cost effective especially if you already own a VCR. The Imager only costs \$195 if you are a HUG member. The Imager is especially cost effective because you don't have to spend over \$20 each for special backup tapes — all you need is high-quality tapes

for your VCR which should cost, at the most, about half of that if you shop around for a good buy. Moreover, a standard 120-minute VHS tape (e.g. T-120) will allow you to record 110MB of programs and data while a typical tape backup system will record only 20-60MB per tape. I highly recommend the Imager as a cost effective and reliable product that can help you ease the backup chore. For additional information about this product, you might want to refer to my "On the Leading Edge" column published in the December 1988 issue of REMark.

One way or another, I recommend that you include some kind of backup software or hardware in your budget if you are planning to buy a hard disk. And when you have a hard disk, an optimizer or "defragmentation" program is also a good investment.

Reorganize the Hard Disk

When you have used your hard disk for awhile, you may notice that the system seems to slow down. Although DOS has a logical way to allocate disk space for files, it essentially amounts to grabbing the first available "opening" (technically called a cluster) for a file. If a file is bigger than a cluster, and many are, then DOS keeps looking for the next available cluster and so on. Because of this file allocation technique, a single file may be "scattered" all over the disk. DOS can still find it with no problem because you may remember that the File Allocation Tables (FATs) contain the CHAIN of clusters for each file. And although this problem can occur with any disk, including a floppy, it becomes more pronounced on a hard disk because of its greater capacity. Consider how the data are read and written by looking at Photo 2.

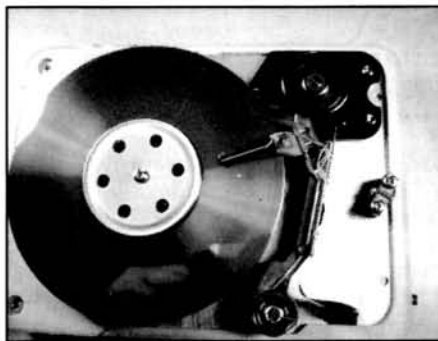


Photo 2
Hard Disk Read/Write Arm Assembly

If you examine Photo 2 closely, you will see that the hard disk's Read/Write heads are all attached to a single movable arm. If a file is scattered in various clusters all over the hard disk, then the arm must move to read or write data. Head movement (i.e. arm movement) is mechanical and it takes time. A lot of head movement translates to a lot of time (comparatively speaking), and as more

and more files have their data scattered, your system will appear to slow down. You can correct this problem by using a utility program that "reorganizes" the hard disk so that the chain for each file contains sequentially numbered clusters. This reorganization — sometimes called optimizing, defragmenting or unfragmenting — reduces the head movement which results in improved system performance. By the way, it is also easier to recover erased files from an "optimized" disk because all of the clusters are in sequential order.

My favorite software for this purpose is the Mace Utilities package that I have mentioned before because of its UNDELETE program to recover erased files. This package also contains the UNFRAG program that will reorganize your hard disk, and I have found it to be extremely safe and reliable. Even though I have never had a problem with this software, it is always good practice to take a complete backup before running any program like this, just in case. You never know when something unexpected, like a power failure, could cause a major problem.

Mace Utilities also contains the SQZD (Squeeze Directory) program that eliminates unused/erased entries from the disk directory. You can also use the SORTD (Sort Directory) program to arrange all file names in each directory or subdirectory in alphabetical order by filename, file type, date or file size. You can also use the REMEDY program to help recover a file on a hard disk that has suddenly developed a bad sector. These are just a few of the extremely useful utilities that you will find in the Mace Utilities package. It is highly recommended.

For Larger Hard Disks

Unless you have Zenith MS-DOS version 3.3 or later, you cannot use a physical hard disk larger than 128MB or have a single partition larger than 32MB. At the time of this writing, I should point out that none of the utility packages available today will work correctly with DOS 3.3 or later, but that will probably be fixed by the time you read this. If you have Zenith MS-DOS 3.3, you will not need to buy a separate utility package anyway because it includes the COMPACT program to optimize a hard disk as well as a utility that can recover erased files. Since you may not have that DOS version yet, one of the points that has been ignored until now is how you work with a hard disk larger than 32MB, say 40MB.

If you are using any release of Zenith MS-DOS 3.2 for a large hard disk, you will have to define partitions no larger than 32MB with the PART command. For the example, it is reasonable to define partitions of 30MB and 10MB (or 20MB and 20MB), and assign the default boot parti-

tion as previously described. Then, FORMAT the hard disk, and load the system software in your bootable partition as before.

The next task is to change the Partition Assignment Flag from "automatic" (the default) to "manual." Use the menu-driven DSKSETUP command to do this. Be sure that you follow these steps exactly in this order because you CANNOT change the partition flag assignment until after the system files have been written to the disk with the FORMAT command. By the way, the Zenith MS-DOS is shipped with the Partition Assignment Flag set to automatic to be compatible with some old PC-DOS software such as TopView or older versions of Microsoft Project. Now you are ready to use the ASGNPART command whose general syntax is shown in Figure 5.

ASGNPART drive-number:partition-number d:

Figure 5
ASGNPART Command Syntax

For a single hard disk, the *drive number* is 0 (zero). The *partition number* is the number of the partition that you saw in the PART command. And *d* is the drive letter as usual. For this example, you would use the following command to assign partition number 2 to drive letter D.

C:\ ==>ASGNPART 0:1 D:

Now you can use the FORMAT program on the partition — FORMAT D:. Once the partition is formatted, you can create subdirectories on it and load any files in the usual way.

Once you know some of the procedures and tricks presented in this article, working with a hard disk is fun and easy. For additional information about hard disks and the commands discussed here, refer to your MS-DOS and owner's manuals.

Next Time

In the next article, we will take a look at some of the other useful DOS commands. If you have a hard disk, you will particularly appreciate the Zenith Search command. We will also look at the file comparison commands -- COMP and FC -- and some other commands that you may find helpful.

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

Products Discussed

HARDWARE

Hard Disk Upgrades
For Z-150 PC series (with controller)
HWD-20 Upgrade Kit (65ms.) \$449.95

For Z-200 and Z-300 containing dual controller (no controller)

HWD-20-AT Upgrade Kit \$379.95
See current catalog for other upgrades and prices

Heath/Zenith Computer Centers
Heath Company
Hilltop Road
St. Joseph, MI 49085
(800) 253-7057 (Heath Catalog orders only)
Imager \$295.00 (\$195 for HUG Members Only)
The Light Pen Company
Box 45255
Los Angeles, CA 90045-0255
(800) 634-1967

SOFTWARE

DSBACKUP+ \$79.95
Design Software
1275 W. Roosevelt Road
Chicago, IL 60185
(800) 231-3088 (except Illinois)
(312) 231-2225 (Illinois only)

Mace Utilities \$99.00
Paul Mace Software, Inc.
400 Williamson Way
Ashland, OR 97520
(800) 523-0258 (Orders only)

*

EGAD

Graphics and Text Screen Print Package for the VGA, EGA, and CGA displays.

- Print any part of the screen ('crop box' lets you use the cursor keys to select any rectangular area)
- Enlarge for emphasis (1 to 4 times in graphics modes).
- Color graphics and text print in color (on color printers) or in black and two shades of gray (for black-only printers).
- Set program configures which screen colors map to which printer colors (or gray and black tones).

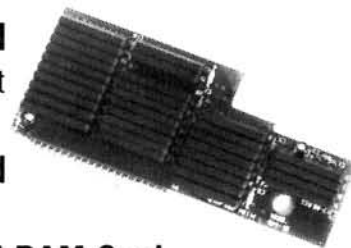
Supported printers: **Epson** and compatibles (Black + 2 grays); **Star NX-1000 Rainbow** (30+ colors); **Xerox 4020** (64+ colors); **Dataproducts 8020** (Black, 6 colors); **NEC-8023/C.Itoh 8510** (Black, 2 grays).
EGAD, \$25.00 Postpaid.

Lindley Systems 4257 Berwick Place,
Woodbridge, VA 22192
(703) 590-8890. Call for Free Catalog

Reader Service #136

Zenith Lap Top Enhancements

- **One Megabyte EMS/Extended RAM Card**-Available for Supersport 286 and Turbosport 386
- **Two Megabyte EMS/Extended RAM Card**-For Supersport 286
- **Four Megabyte EMS/Extended RAM Card**-For Supersport 286
- **ARC net LAN Card**
For Supersport 286
- **Streamer Tape Back Up System**-For Supersport 286
- **Two Megabyte CMOS RAM Disk**-For Z181



One Year Warranty and 30 Day Return.

For Information:
CALL 714-540-1174
FAX 714-540-1023
or Write

Zenith is a trademark of
Zenith Data Systems

Ai AMERICAN
CRYPTRONICS INC.

1580 Corporate Dr., Suite 123
Costa Mesa, California 92626

Reader Service #101

QUIKDATA - 12 YEARS OF H/Z SUPPORT!

For all your H/Z 8-bit and PC/XT/AT needs

ACCELERATE YOUR PC!

From Sota Technologies, Inc., the fastest and most proven way to really speed up your Heath/Zenith PC/XT computer, giving it -AT compatible speeds! Turn your turtle into a -286 rabbit with a 12.5 Mhz 80286 accelerator board.

There are other ways to speed up your H/Z PC/XT computer, but you spend too much and get too little. The **286i** is the effective solution, making your H/Z150/160/150/159 series of computers, or any general PC/XT computer in many cases faster than a standard IBM AT type computer! Simple half-card plug-in installation with switchable speed control.

Complete with 16-bit -AT -286 processor, 16K on-board CACHE memory for dramatic speed increase, 12.5 Mhz 80286 processor, and software. **You won't believe your stop watch!**

EXP-12 - \$379

MEMORY UPGRADES

Note: All memory upgrades come without memory chips. Call for current chip pricing.

Z150MP - \$19 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.

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 Allows you to upgrade your H/Z110/120 (new motherboards with p/n 181-4918 or greater) to 768K system RAM. Requires 27 256K DRAM chips.

WINCHESTER UPGRADE KITS

PCW20 - \$295 Complete winchester setup for a H/Z150, 148, 158, 159, 160, PC etc. Includes 21 meg formatted half-height Segate ST-225 65ms drive, WD WX1 winnie controller board, cable set, documentation.

PCW30 - \$380 32 meg with 38ms Segate ST-138.

PCW40 - \$449 42 meg with 37ms Segate ST-251.

PCW80 - \$695 80 meg with Segate ST-4096 full size drive.

EWIN20 - \$555 Complete 21 meg winchester package for the Z100. Includes S100 host adapter card, WD winchester controller, Segate ST-225 winchester drive, cables, software and documentation. We also have this in 32 megs (\$640), 42 megs (\$689), and 80 megs.

We also have winchester systems for the H8 and H/Z89/90.

BARE WINCHESTER DRIVES

ST-225 - \$229 65ms half height bare winchester with 21 megabyte storage capacity.

ST-251 - \$395 37ms autopark half height bare winchester with 42 megabyte storage capacity. Good choice for -AT compatibles.

ST-251-1 - \$429 Same as ST-251 but 28ms speed.

ST-4096 - \$629 28ms autopark full height bare winchester with 80 megabyte storage capacity. Best choice for -AT compatibles.

ACCELERATE YOUR 241/248

With the Aox Z-MASTER **20Mhz 80386**, your Z241/248 computer will be faster than the Zenith Z386 and more compatible. The Aox 20Mhz Z-MASTER 386 is a one-board CPU card that replaces the Zenith 80286 CPU board of the Z241 and Z248. With support for up to 8MB of onboard 32-bit memory and 80387 math co-processor on one single card. The Z-MASTER 386 comes standard with an efficient 8KB cache providing 0 wait state access to the onboard 32-bit memory and optimum access to existing 16-bit Zenith and conventional memory. The Phoenix ROM BIOS Plus is also standard providing a higher level of software and hardware compatibility. All existing 16-bit memory is accessed efficiently via the onboard cache controller. Installation is simple with only one board to replace, approximate installation time is only 10 minutes. See H-SCOOP #107, or request a copy for a report on this board, or inquire for more information.

AOX386 - SPECIAL - \$1295 0K RAM

APX386-1 - \$1695 with 1MB RAM

PRINTERS

We have a full variety of Panasonic printers; narrow and wide carriage dot matrix, 24-pin dot matrix and daisywheel. We also have cables, buffers, ribbons, tractors when not included, etc:

KXP-1091i	80 column 160 cps parallel std	\$225
KXP-1092i	80 column 240 cps parallel std	\$359
KXP-1595	132 col 240cps ser/par std	\$489
KXP-1524	132 col 240 cps 24pin par std	\$579
KXP-1124	80 col 192 cps 24pin parallel	\$369
KXP-3151	132 col daisy 22cps par std	\$539

8-BIT

We carry a full line of hardware and software products for the H8/H89/90 computers. This includes diskettes, printers, H89 working boards and parts, some H8 parts, etc. Call or write to obtain a catalog of all our 8-bit goodies.

H37 SOFT SECTOR CONTROLLER	\$179
H8 SOFT SECTOR/WINNIE CONTROLLER PACKAGE	\$275
TM100-2R Tandon DS ST 48 TPI refurb disk drive	\$79
H89 4MHZ - Double your H/Z89/90 CPU speed	\$43
Z47 REMEX DRIVES slaves and masters - special	\$95

H-SCOOP

Of course, don't forget about the best source of Heath/Zenith related information you can obtain - our monthly newsletter, **H-SCOOP**! Just \$24 for a 12-issue year (\$35 foreign), it will help you get the most from your computer investment. Get sound technical advice, helpful hints, find out what the problems are, fixes, reports, reviews, information from other subscribers, classifieds and much more.

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

QUIKDATA, INC.

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

A Z-100 Assembly Language Program to Send Setup Codes to the C. ITOH Prowriter Parallel Printer

Les Landers
2841 Latham Drive
Sacramento, CA 95864

If you have ever wished for an easy way to send setup codes to your printer directly from Z-DOS or MS-DOS, then this little program may be just what you need.

This article was spurred by the repeated admonitions I've seen in print that we Z-100 owners should try our hands at writing supporting articles if you want to see more space in print devoted to our trusty old (!) machines. I hope this helps to do that.

Assembly language piqued my curiosity when I built my low-profile H-100, joined HUG, and began to receive REMark with all those funny-looking listings. Soon I realized that here was a way to do things that were difficult or nearly impossible to do with BASIC. I bought the Heath Macro 86 Assembly Language Programming course, and later I bought "The Serious Assembler" by Crayne and Girard. I recommend this little book as a good "how-to" supplement to the Heath course; it is written for PC-DOS programmers, but much of the material is equally applicable to the Z-100 and other generic MS-DOS computers.

By working with these books and studying the examples given in REMark and other publications, I've managed to barely reach the point where I feel comfortable enough with assembly language to attempt short utility programs such as PRINTER.COM to meet my needs. I wrote it mainly as a "learn by doing" exercise in Assembly language, and since then I have found it to be a frequently-used time saver. I offer it as a useful utility, as well as an instructive example of simple Assembly language programming techniques.

What the Program is: Application programs and programming languages generally give the user a way to send setup control characters to a printer, but I frequently found this to be inconvenient. I wrote PRINTER.COM so I could send setup

codes to my Prowriter printer directly from DOS. If you use a parallel printer other than the Prowriter, read on: Converting the program to your needs should be a simple job.

I frequently use it to set up my printer before using the MS-DOS COPY or PRINT command to print out a text file. I also use it to set the printer to unidirectional mode when printing a Doodler .PIC file; this eliminates the slight line-to-line offset that is usually negligible when printing text but which spoils the appearance of a picture.

What it Does: When PRINTER.COM is invoked, it displays a menu on the screen from which the user can select single keystrokes to send setup codes to the C. Itoh Prowriter parallel printer. The keystrokes are echoed to the screen, making it easy for the user to keep track of which setup codes have been sent to the printer. When all selections have been made, a single keystroke causes the program to exit to DOS. If the printer status is Not Ready when the program is invoked, the program exits to DOS immediately.

How it Works: The first few lines in Listing 1 take care of the housekeeping necessary to set up a .COM program: The entire program is defined as a single segment named COMSEG, the ASSUME directive is used to tell the assembler that all four segment registers will point to it, and the ORG directive sets the load point as offset 100H from the start of the Program Segment Prefix.

The first instruction, at the label START:, causes a jump over the data area to the first executable code. The data definition area is included up front so that the assembler can process all these items before it encounters any code that refers to them.

The main program, which begins at the label ENTRY:, starts by sending the Z-100 "Clear Screen" escape sequence to the console, using the subroutine defined at the label DISPLAY. Using a procedure as a subroutine in this way simplifies the

code-writing when the same set of commands is used repetitively in a program.

Next, the program tests whether the printer status is Ready by the method described in REMark May 1986 ("Printer Ready? A Solution to a Z-100 Hangup," by N. B. Day). It does this by reading the byte at port address 0E2 hex, AND'ing it with 1 to isolate the lowest-order bit (the parallel printer BUSY line), and testing whether the result is 0 or 1.

If the result is 0 (low), the printer is on-line and ready to accept characters. If the result is 1 (high), the printer is not ready, either because it is really busy or because it is turned off (in this case, the circuitry of the Z-100 has pulled the BUSY line high).

If the printer status is Ready, the program jumps to the label READY: and prints the user menu shown in Figure 1, highlighted in reverse video. If the printer status is Not Ready, the program sounds a warning beep, prints an error message highlighted in reverse video, and exits to DOS.

The section of code beginning at the label KEY: uses Interrupt 21H, Function 8, which waits for a character to be entered at the keyboard and then places the character in Register AL without echoing it to the screen.

Each time a character is entered at the keyboard, the program tests the contents of Register AL to determine whether the character is the menu character for selecting Pica pitch. If it is, the character is first echoed to the screen using the subroutine defined at the label ECHO and then the Z-100 escape sequence for setting the Prowriter to Pica pitch (ESC 78) is sent to the printer one character at a time. This is done with Interrupt 21H, Function 5, which sends the character in Register DL to the parallel device. Finally, the program recycles back to the label KEY: to wait for another character to be entered.

If the character entered at the keyboard is not the menu character for selecting Pica pitch, the program jumps to

the next section and tests whether the character entered at the keyboard is the menu character for selecting Elite pitch. If it is, the character is echoed to the screen, the Z-100 escape sequence for setting the Prowriter to Elite pitch (ESC 69) is sent to the printer, and the program recycles to the label KEY: to wait for the next character.

If the character entered at the keyboard is not the menu character for selecting Elite pitch, the program jumps to the next section, and so on.

In this way, the program tests each character entered at the keyboard, looking for a match with a valid menu character. If a match is found, the corresponding menu action is taken, either by sending a setup code to the printer or by exiting to DOS. If no match is found in the last section, at the label EXIT:, the program beeps a warning and recycles and waits for another character to be entered. This process continues until the menu character for exiting to DOS is entered, whereupon the program prints a completion message and exits to DOS.

The line of code at the label EXIT: sets the third-highest-order bit of the number in register AL to zero. If the number in AL is the ASCII value for an uppercase letter, nothing happens. However, if AL contains the ASCII value of a lowercase letter, this converts it to up-

percase by subtracting 32. If the contents of AL are not the ASCII value for a letter, no harm is done because the program will ignore it and recycle to wait for the next keystroke.

Assembling: Type Listing 1 with an editor that produces true ASCII files (such as WatchWord, PeachText, or WordStar in the non-document mode). Assuming that MASM.EXE, LINK.EXE, and EXE2BIN.EXE are on the default drive together with PRINTER.ASM, create PRINTER.COM by issuing the following commands:

```

MASM PRINTER; <RETURN>
LINK PRINTER; <RETURN>
EXE2BIN PRINTER.EXE PRINTER.COM <RETURN>
ERASE PRINTER.OBJ <RETURN>
ERASE PRINTER.EXE <RETURN>

```

Remember that you can ignore the 'No Stack' warning given by the LINK program, since this is going to be a .COM program.

The program runs under any version of Z-DOS or MS-DOS, and the source file assembles, links, and converts okay with all versions of MASM, LINK, and EXE2BIN I've tried, from the versions included free with Z-DOS (the good old days!) up to the versions in the MS-DOS Version 3 Programmer's Utility Pack. This is as expected, since nothing in the program requires a DOS version higher than 1.xx.

Miscellaneous: The program is reasonably bullet-proof. If the printer status

is Not Ready when the program is invoked, the program beeps a warning, displays an error message on the screen, and exits to DOS rather than hanging the system. If the user enters an invalid keystroke (one not on the menu), the program simply beeps a warning and waits for the next keystroke without echoing the invalid keystroke to the screen. If the user makes an erroneous choice from the menu, the mistake can be corrected simply by striking the desired key before exiting the program. For example, if you strike the Pica key when you really want Elite, simply strike the Elite key to override the Pica selection before striking X to exit.

However, if the printer status changes from Ready to Not Ready during execution of the program, striking a valid menu key will cause the system to hang until the printer status returns to Ready. I've made this happen by deliberately taking the printer off-line or by turning it off after invoking the program, but in each case, normal system operation was restored when I put the printer back on line.

Modifications: The hangup condition described above MIGHT be avoided by expanding the program to include a separate test to determine whether the printer is ready before each character is sent. This occurred to me after I'd finished the program, but I didn't bother to make the change because the condition is not very likely to occur and the consequences are not serious.

The program as presented here includes the relatively small number of Prowriter setup control codes that I routinely use. Additional codes could easily be added to suit the user's needs. Also, the program could easily be modified for using the Z-100 with a different parallel printer by replacing the ProWriter codes with the appropriate setup codes for that printer.

References Cited

"Macro 86 Assembly Language Programming," Heath Part No. EC-1201 Charles A. Crayne and Dian Girard, "The Serious Assembler," Baen Publishing Enterprises, 260 Fifth Avenue, New York, NY 10001, 1985, ISBN 0-671-55963-X.

Most home finance programs have serious limitations.

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

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

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

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

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

Only \$49⁹⁵

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

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

Reader Service #137



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

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

```

* C. Itoh ProWriter Parallel Printer Setup - Touch Keys to Make Selections
*
1 - Pica Pitch (10 char/inch)      5 - Set Boldface Print
2 - Elite Pitch (12 char/inch)     6 - Clear Boldface Print
3 - Compressed Pitch (17 char/inch) 7 - Unidirectional Print
4 - Proportional Pitch            8 - Bidirectional Print
*
* X - Exit Printer Setup Program
*

```

Figure 1
PRINTER.COM Selection Menu

Listing 1

TITLE PRINTER.ASM

; Z-100 Program to send control codes to the C. Itoh ProWriter printer.

; Program written September 1986, modified August 1988 by L. C. Landers

; The following settings can be selected from a menu:

; Pitch: Pica (10 cpi), Elite (12 cpi), Compressed (17 cpi), or

; Proportional

; Bold Print: Set or clear

; Print Direction: Unidirectional or Bidirectional

; Get things organized

; COMSEG SEGMENT

ASSUME CS:COMSEG, DS:COMSEG, ES:COMSEG, SS:COMSEG

ORG 100H

START: JMP ENTRY ;Skip over data definition area

; Data definition area

; CLRSN DB 27, 'E', '\$' ;H/Z-100 'clear screen escape seq

; BEEP DB 7, '\$' ;ASCII BEL

; ERRMSG DB 27, 'p' ;Start reverse video

; DB 'PRINTER NOT READY TO RECEIVE CONTROL CODES'

; DB 27, 'q' ;Stop reverse video

; DB 13, 10, '\$'

; ENDMSC DB 13, 10, 10 ;CR + LF twice

; DB 27, 'p' ;Start reverse video

; DB 'PRINTER SETUP COMPLETE'

; DB 27, 'q' ;Stop reverse video

; DB 10, '\$'

; DB 27, 'p' ;LF

; DB '*, 76 DUP ('-', '*', 13, 10 ;Start reverse video

; DB 'C. Itoh ProWriter Parallel Printer Setup -

; DB 'Touch Keys to Make Selections', 13, 10

; DB '*, 76 DUP ('-', '*', 13, 10

; DB '1 - Pica Pitch (10 char/inch)

; DB '5 - Set Boldface Print', 13, 10

; DB '2 - Elite Pitch (12 char/inch)

; DB '6 - Clear Boldface Print', 13, 10

; DB '3 - Compressed Pitch (17 char/inch)

; DB '7 - Unidirectional Print', 13, 10

; DB '4 - Proportional Pitch

```

DB '8 - Bidirectional Print', 13, 10
DB '*, 76 DUP ('-', '*', 13, 10
DB 'X - Exit Printer Setup'
DB 'Program', 13, 10
DB '*, 76 DUP ('-', '*', 13, 10
DB 27, 'q', '$' ;Stop reverse video

```

;Start of main program

ENTRY: MOV DX, OFFSET CLRSN ;Start by clearing the screen

CALL DISPLAY

IN AL, 0E2H ;Read byte at Port 0E2H

AND AL, 0000001B ;Isolate lowest order bit

JZ READY ;Jump if FALSE (printer is ready)

MOV DX, OFFSET ERRMSG ;Otherwise, the printer is NOT ready,

CALL DISPLAY ;so display error message

MOV DX, OFFSET BEEP ;

CALL DISPLAY ;and sound a beep

JMP QUIT ;and exit to DOS.

MOV DX, OFFSET MENU ;Display the menu

CALL DISPLAY

MOV AH, 08H ;DOS 'Keybd char, no echo' function

INT 21H ;Wait for keyboard char, put in AL

CMP AL, '1' ;Pica selected?

JNE ELITE ;Jump if no

CALL ECHO ;Echo character to screen

MOV AH, 05H ;DOS 'Character to printer' function

MOV DL, 27 ;ASCII ESCape character

INT 21H ;Send ESC to printer

MOV DL, 78 ;C.Itoh code for Pica

INT 21H ;Send code to printer

JMP KEY ;and recycle

CMP AL, '2' ;Elite selected?

JNE COMPR ;Jump if no

CALL ECHO ;Echo character to screen

MOV AH, 05H ;DOS 'Character to printer' function

MOV DL, 27 ;ASCII ESCape character

INT 21H ;Send ESC to printer

MOV DL, 69 ;C.Itoh code for Elite

INT 21H ;Send code to printer

JMP KEY ;and recycle

CMP AL, '3' ;Compressed pitch selected?

JNE PROFOR ;Jump if no

CALL ECHO ;Echo character to screen

MOV AH, 05H ;DOS 'Character to printer' function

MOV DL, 27 ;ASCII ESCape character

INT 21H ;Send ESC to printer

MOV DL, 81 ;C.Itoh code for Compressed mode

INT 21H ;Send code to printer

JMP KEY ;and recycle

CMP AL, '4' ;Proportional pitch selected?

JNE BOLD ;Jump if no

CALL ECHO ;Echo character to screen

MOV AH, 05H ;DOS 'Character to printer' function

MOV DL, 27 ;ASCII ESCape character

INT 21H ;Send ESC to printer

MOV DL, 80 ;C.Itoh code for Proportional mode

INT 21H ;Send code to printer

JMP KEY ;and recycle

CMP AL, '5' ;Boldface selected?

*

Classified Ads

WILL GIVE GOOD HOME TO A HERO. Children's museum wants robot. Fully assembled, partially, or still in box. Donations accepted or must be cheap. Bill Dreadin (904) 651-5097 days/(904) 837-6993 nights.

Z-319 BIT MAPPED GRAPHICS CARD with software. \$95.00. (206) 252-0549.

COMPUTERS AND ROBOTS: Laptops-Toshiba T3100/20 meg 286 \$2,900. TS100/40 meg 386 \$4,800. Full warranty, Epson HX20 notebook computer \$600. Zenith H/Z-100 multi-compatible with plotter. Robots from \$25 to \$950. Yool, 2060 S. Val Vista, Apache Jct., AZ 85219. (602) 982-7263.

COMPUTER PROGRAMS: AUDIO OR VIDEO LIBRARY \$29.95 each, \$49.95 both. Biorythm \$7.50, Astrology, \$7.50, \$12.00 both. Yool, 2060 S. Val Vista, Apache Jct., AZ 85219. (602) 982-7213.

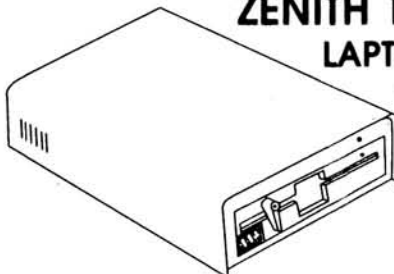
;Last wrapup
;COMSEG
ENDS
END
START



WELTEC

W 525 Subsystem

5 1/4" EXTERNAL DRIVE FOR ZENITH 181 & 183 LAPTOP COMPUTERS



- EXTERNAL 5 1/4" DRIVE FOR ZENITH 181 & 183
- ACTS AS ADDITIONAL SYSTEM DRIVE
- TRANSPORT AND SHARE DATA WITH PCs
- POWER SUPPLY & CABLE INCLUDED
- EXTERNAL DRIVES FOR OTHER LAPTOPS AVAILABLE

WELTEC

17981 Sky Park Circle, Suite M

Irvine, CA 92714

(714) 250-1959

Reader Service #194

```

JNE NOBOLD
CALL ECHO
MOV AH, 05H
MOV DL, 27
INT 21H
MOV DL, 33
INT 21H
JMP KEY
JMP AL, '6'
JNE UNI
CALL ECHO
MOV AH, 05H
MOV DL, 27
INT 21H
MOV DL, 34
INT 21H
JMP KEY
JMP AL, '7'
JNE UNI
CALL ECHO
MOV AH, 05H
MOV DL, 27
INT 21H
MOV DL, 62
INT 21H
JMP KEY
JMP AL, '8'
JNE UNI
CALL ECHO
MOV AH, 05H
MOV DL, 27
INT 21H
MOV DL, 60
INT 21H
JMP KEY
JMP AL, 01011111B
JNE NOTE
JMP AL, 'X'
JNE NOTE
MOV DX, OFFSET BEEP
CALL DISPLAY
JMP KEY
JMP DX, OFFSET ENDMG
CALL DISPLAY
QUIT: INT 20H
;
;Subroutine definitions
;
DISPLAY PROC NEAR
MOV AH, 9
INT 21H
RET
DISPLAY ENDP
ECHO PROC NEAR
MOV DL, AL
MOV AH, 02H
INT 21H
RET
ECHO ENDP
;
;Copy keyboard character to DL
;DOS 'Character to screen' function
;Write character in DL to screen

```



MailMerge Rosters and Mailing Lists

Part III

Ralph E. Camp
13331 Hanford-Armona Road
Hanford, CA 93230

Having completed the part of the system for the mailing list, we move now to the part of the system for the roster. Figure 5 is the new roster, the file CHURCH.DAT

sorted by denomination. Again, you must decide how you want your information to be sorted, if at all, and how it is to be printed.

Figure 5

The New Roster, Sorted by Denomination

Note: For illustration purposes only, the pages have been cut to two addresses, and a second page has been made to display the HEARER.

Roster

This roster is supplied to members for their own use, and is not for general distribution, or to be given to, or used by, outside organizations as a mailing list for solicitation.

Assembly of God
Pastor: Rev. George Goodfellow
2992 Apple Street
Lemoore, CA 93245

OFFICE 924-2345
HOME 924-3456

First Baptist Church
Pastor: Dr. Sam Shepherd
321 Jump Street
Lemoore, CA 93245

OFFICE 924-6789
HOME 924-7890

ROSTER

page 2

Lemoore Christian Aid Society
Director: Mrs. Jane Doe
1001 East West Avenue
Lemoore, CA 93245

OFFICE 924-5432
HOME 924-6543

Lemoore Christian Servicemen's Center
Director: Rev. Ralph Camp
13331 Hanford-Armona Road
Hanford, CA 93230

OFFICE 584-0436
HOME 584-0436

It is a two step process to print this roster. The first step writes the formatted data to a disk file. The second step prints that file to paper.

With your WordStar program at the OPENING MENU, open a DOCUMENT FILE. I named this program SETROSTR.PGM because it sets up the format or layout of the roster.

The second non-printing comment states that the computer should be commanded to put this information on another file on the disk instead of sending it to the printer. This two step process is necessary because this program does not provide for any space at the top or bottom of a sheet of 11 inch paper, nor does it allow for inserting any other text on the finished

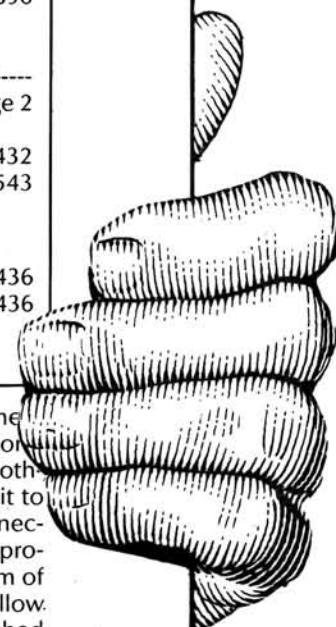


Figure 6
The Program to Print the Roster Format to a Disk File

```

L-----!-----R
..This file is called SETROSTR.PGM <
..This program is to be used to print to a disk file ROSTER.FRM<
..It is to format a roster from CHURCH.DAT <
..Typed (date) <
.PL 6 M
.MT 0 M
.MB 0 M
.PO 0 <
.PF OFF M
.DF CHURCH.DAT M
.RV SORTNAME, ADDSTREET M
.RV CHURCH, ADDMAIL, CITY, STATE, ZIP, OFFICEPH M
.RV LASTNAME, FIRSTNAME, POSITION, TITLE, HOMEPH M
&CHURCH& -
                                OFFICE &OFFICEPH& <
&POSITION&: &TITLE& &FIRSTNAME& &LASTNAME& -
                                HOME &HOMEPH& <
..IF &ADDMAIL& = "(none)" GOTO M
(mail to) &ADDMAIL& <
.EF M
&ADDSTREET& <
&CITY&, &STATE& &ZIP& <
.PA <
-----P

```

page. By using this two step process, it is possible to have a custom formatted finished page with any other text necessary.

The first dot command is the PAGE LENGTH (.PL). This is not the 66 lines of an 11 inch page, but the number of lines from the top of one address to the top of the next address printed on that page. I use six lines because that is what fits my needs. Let me suggest that unless you need more than five lines for your addresses, that you use .PL 5. This will give you 11 addresses on an 11 inch page, or 55 lines of type. The default or standard setting for WordStar is 55 lines of type on an 11 inch page, so you will not have to edit the finished file to add non-standard page breaks.

MARGIN TOP (.MT), MARGIN BOTTOM (.MB) and PAGE OFFSET (.PO) are all zero as in the previous programs. Be sure you have PAGE OFFSET set to zero for this first step or you will start printing in the middle of the page for the finished print.

A new dot command that has not been used up to this point is PRINT TIME LINE FORMING (.PF). This is the feature of WordStar that reforms the text of a page when text is added to or deleted from a line or paragraph. In this special case, this feature needs to be turned OFF to allow the computer to use the specific, non-standard format for your individual need.

The DEFINE FILE (.DF) and READ VARIABLES (.RV) dot commands are the same as in your previous print programs.

You are now ready for the first line of your address. In the illustration of Figure 5, the first line of address is the church's name and office phone number. The second line is the person's name and home phone number. In typing this part of the

program, it is very important to follow each step exactly, or the program will not work properly. This is the experience of several hours of frustration speaking.

1. Determine at what column the second generic name will print, and set the tab stop accordingly. Translation: Decide how far over you want the second piece of information (in this case, the office phone number) to start printing, and set the tab stop to that column. This is explained in steps two and three.
2. Type Control O N (^O N) and at the top of the screen will be the prompt: CLEAR TAB AT COL (ESCAPE for cursor col; A for all)? Enter the letter "A" to clear all of the tabs.
3. Now type Control O I (^O I). After the prompt appears, type the column number you want to use. In the case of the illustration, it is column 46. This is followed by a RETURN. You should now see only one exclamation point (!) appearing in the ruler line, and that at the column you have specified. Look at the top of Figure 6.
4. Now type the generic name for the information that goes on the left side of the page. It is possible to have more than one, as in the second line of the illustration.
5. Do not put or leave any blank spaces after the last ampersand (&). The generic names(s) for the left side of the page are followed by a Control P (^P) and RETURN. Note that a hyphen (-) appears on the right side of the screen instead of a carriage return mark (<) in the line ended with the ^P RETURN. Also, the line count in the ruler line does not advance.

6. Press the TAB key or use Control I (^I) to move the cursor to the tab stop. DO NOT USE THE SPACE BAR TO GET THE CURSOR TO THE TAB POSITION!!! The WordStar manual is very specific about this, and my experience confirms it.

7. Type the generic name(s) for the data to be printed starting at the tab set. This is followed by a RETURN without the Control P (^P).

8. Line one is now complete. Move to line two. Follow steps four through seven for each line that uses a tab set in it. It is possible to space over a few spaces to the right from the tab set, as in line two of the illustration, but never use the space bar to get to the tab position from the left side of the page.

Now comes the .IF and .EF dot commands. There is only one set in this program because I want the street address printed, and also the mailing address IF there is a separate one.

The rest of the program is like the mailing program. Don't forget the PAGE command (.PA) and the last RETURN at the end. Type Control K D (^K D) and return to the OPENING MENU.

It's test time again. With your WordStar program at the OPENING MENU, type "M" to Run MailMerge. At the prompt, type the program name. In this illustration, it is SETROSTR.PGM. This time, the program name must be followed by a RETURN, because you need to answer the next set of questions. The first question you will see will be: Disk file output (Y/N): The default is "N", but you type "Y" because you do want a disk file. The next question will be: Output file name? In the illustration, the name is ROSTER.FRM for the FORM the ROSTER is to be printed in. You can now press the ESCAPE key to skip the rest of the questions.

If you have run this program before and the file ROSTER.FRM is still on the disk, you will get a prompt: File A:ROSTER.FRM exists -- overwrite? (Y/N): If you are making a new copy of your roster because of changes and corrections, you will want to overwrite the old file. All this does is replace the old copy of the file with a new copy.

You may feel that you, or someone else you are setting up the system for, will not remember to command SETROSTR.PGM to write to a disk file. Figure 10 is a fancy control program you can use to remind yourself and/or others to instruct the computer to write to a disk file instead of the printer. You can copy it just as it is, if you want to. I first wrote this for myself, because I kept forgetting to write to disk. However, if you do use this control program, be sure to make the changes to the top of the first program as shown in Figure 9, and be sure to make the name change on the file directory as well.

I also need to print a phone listing

Figure 7
The Program to Print the Phone List to a Disk File

```

L-----!-----R
..This file is called NAME.FLE <
..It can be run by the program file SETPERSN.PGM <
..This program is to be used to print to a disk file PERSON.FRM<
..It is to format a phone list from PERSON.SRT <
..Typed (date) <
..PL 3 M
..MT 0 M
..MB 0 M
..PO 0 <
..PF OFF M
..DF CHURCH.DAT M
..RV SORTNAME, ADDSTREET M
..RV CHURCH, ADDMAIL, CITY, STATE, ZIP, OFFICEPH M
..RV LASTNAME, FIRSTNAME, POSITION, TITLE, HOMEPH M
&POSITION&: &TITLE& &FIRSTNAME& &LASTNAME& -
HOME &HOMEPH& <
&CHURCH& -
OFFICE &OFFICEPH& <
..PA <
-----P

```

sorted alphabetically by the person's name. Using my sort program, I sorted the file CHURCH.DAT and named the sorted file PERSON.SRT. Figure 7 is the program to write that sorted data to a disk file. It is run by the program SETPERSN.PGM, which is a control program like SETROSTR.PGM in Figure 10.

The appropriate changes have been made in the non-printing comments at the top of the program, and the file to be read is changed to PERSON.SRT. The PAGE LENGTH is now three instead of six. Lines one and two have been reversed using the Control K V (^K V) command, and the rest of the address has been deleted using the Control K Y (^K Y) command. That program is finished. It was not a lot of work after doing the first one. Time to give it a try.

One final print program to go, and the roster will be complete. This is the print program that takes the disk file ROSTER.FRM and the disk file PERSON.FRM and prints them on paper. Most people will not need the second file for the phone listing.

Actually, you can go one of two ways. You can either edit the file ROSTER.FRM and print it, or you can write the print program. For myself, I use the print program, because I do not want to retype the opening information at the top of the file each time I have changes and/or corrections. Also, I am printing two different files in one printing. The information in this data file does not change a lot, so the opening text could be retyped each time. However, another file I have changes very frequently. It is also possible that you may have much more opening text than these three lines. Another possibility is to type the text into its own file, and then each time it is needed, read that file into the roster file using the Control K R (^K R) command.

lines with the dot command .MB 9, I decrease the total text to 54 lines.

If you are using five lines for each of your addresses, as I suggested earlier, you will not need to make this adjustment because you will have 11 addresses of five lines each, or 55 total line of text per page. However, you may need still a different spacing. Use what you need for your program.

Since I do have text at the top of my roster, I have centered the word "ROSTER" on text line one. This is followed by the dot command HEADING (.HE). Notice that the word "page" extends beyond the right margin. However, when the program is printed, the page number will be flush with the right margin. When typed this way — the heading after the first line of type — the heading will not be printed until the second page. The heading will then be printed in the top margin on page two and every page following. Also, the correct page number will be in-

Figure 8
The Program to Print the Roster on Paper

```

..This file is called PNTRSTR.PGM <
..It is to print the files ROSTER.FRM and PERSON.FRM on paper <
..The programs SETROSTR.PGM and SETPERSN.PGM must be run first <
..Typed (date) <
..MB 9 <
ROSTER <
This roster is supplied to members for the their own use, and is <
not for general distribution, or to be given to, or used by, <
outside organizations as a mailing list for solicitation. <
..FI ROSTER.FRM M
..PA <
-----P
..FI PERSON.FRM M

```

Notice there is only one dot command before the text in this program, and that command is MARGIN BOTTOM (.MB). All of the rest of the page layout commands will follow the default or standard spacing. Since I am using six lines for each address, I can get nine addresses on a page, for a total of 54 lines of type. WordStar default or standard page layout is 55 lines of type, or one more than I need. The default or standard margin at the bottom of a page is eight lines. By increasing the margin to nine

sorted for the pound sign (#) after the word "page."

Now type the opening text. The only "have to" to keep in mind is to have your FILE INSERT (.FI) command to start on the first line of where an address would normally start. In this illustration, where each address is six lines of type, it will be line seven, line 13, line 19, etc. For five line addresses, it will be line six, line 11, line 16, line 21, etc.

If you do not have text to print at the top of your roster, I suggest you simply

Figure 9
The Top of the Program to be Run by SETROSTR.PGM

These are the changes to be made to the file SETROSTR.PGM if the control program is used. The control program becomes SETROSTR.PGM and this file becomes FORMAT.FLE. Be sure to make the changes on the disk file directory also.

```

..This file is called FORMAT.FLE <
..It is normally controlled by the program file SETROSTR.PGM <
..This program is to be used to print to a disk file ROSTER.FRM<
..It is to format a roster from CHURCH.DAT <
..Typed (date) <

```

A-One H/Z Enhancements!

Clock Uses No Slot

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 \$35.00, Z-100 \$36.50 Module: \$27.50

Configuration Control

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

H/Z-148 Expansions

ZEX-148: Adds one full-size and one half-size expansion card slot. \$79.95

ZP-148: Replacement PAL chip expands existing 640K memory to 704K. \$19.95

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

VCE-150: Eliminate video card. Install EGA or VGA card. All plug in. \$39.95, SRAM Chip (Required) \$20.00

RM-150: PROM used in removing video card. With detailed instructions. \$9.95

ZP640 PLUS: Expand standard memory card to 640/704K with 2 banks of 256K RAM chips (not included). \$19.95

LIM150: Get 640K RAM plus 512K of simulated Lotus/Intel/Microsoft EMS v3.2 expanded memory. Installs on standard memory card. No soldering. Must have 45 256K RAM chips (not included). \$39.95

MegaRAM-150: Get 640/704K plus 512K RAM disk on standard memory card. No soldering. Without RAM chips. \$39.95

COM3: Change existing COM2 address. Put internal MODEM at COM2. Don't lose serial port. Includes software. \$29.95

Maximize Your Z-100

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. \$49.00

Z-171 Memory Expansion

MegaRAM-171: Put 256K RAM chips (not included) on existing memory card. Get 640K plus 384K RAM disk. \$59.95

H/Z-89 Corner

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, phone, or see your Heath/Zenith 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

Figure 10
A Command Program to Run FORMAT.FLE

```
..This file is called SETROSTR.PGM <
..It is a command file program to ensure that FORMAT.FLE is <
..printed to a disk file named ROSTER.FRM <
..Typed (date) <
.DM THIS PROGRAM IS TO OUTPUT TO THE DISK FILE <ROSTER.FRM> M
.DM M M
.DM IF YOU HAVE SET THE PROGRAM CORRECTLY, M
.DM TYPE <Y> FOLLOWED BY A <RETURN>. M
.DM M M
.DM IF THE PROGRAM IS NOT SET CORRECTLY, M
.DM TYPE <N> FOLLOWED BY A <RETURN>. M
.DM M M
.AV 'IS THE PROGRAM SET CORRECTLY? (Y/N): ', QUESTION, 1 M
.EX &QUESTION& = 'Y' .OR. &QUESTION& = 'y' GOTO ABORT M
.CS M
.DM ***** PROGRAM WORKING ***** M
.FI FORMAT.FLE M
.CS M
.DM ***** PROGRAM COMPLETED ***** M
.EF ABORT M
.IF &QUESTION& = 'Y' .OR. &QUESTION& = 'y' GOTO END M
.CS M
.DM ***** PROGRAM ABORTED ***** M
.DM M M
.DM RESTART THE PROGRAM AND INSTRUCT THE COMPUTER TO M
.DM OUTPUT TO DISK, AND NAME THE DISK FILE <ROSTER.FRM> M
.EF END M
```

Figure 11
The Program to Print the Roster on 2 X 4 Index Cards

```
..This file is called INDEXADD.PGM to print CHURCH.DAT <
..This is to print on 2 X 4 inch rotary file card <
..Typed (date) <
.PL 13 <
.MT 0 <
.MB 0 <
.PO 0 <
.DF CHURCH.DAT M
.RV SORTNAME, ADDSTREET M
.RV CHURCH, ADDMAIL, CITY, STATE, ZIP, OFFICEPH M
.RV LASTNAME, FIRSTNAME, POSITION, TITLE, HOMEPH M
&CHURCH& <
.IF &ADDMAIL& = '(none)' GOTO <
(mail to) &ADDMAIL& <
.EF M
&ADDSTREET& <
&CITY&, &STATE& &ZIP& <
Office phone &OFFICEPH& <
&POSITION&: &TITLE& &FIRSTNAME& &LASTNAME& <
Home phone &HOMEPH& <
.PA <
-----P
```

edit the file and print it. Change the MARGIN BOTTOM (.MB) if you need to, and add whatever heading you want, and print.

For most people, the last line of this program will be the FILE INSERT (.FI) command and a RETURN. There is not a final page command (.PA) for this program because this program ends with the end of the inserted file. However, since I also need to print the phone listing as a part of this roster, I have two more commands. To make the computer start printing the phone list on its own page, a page break

(.PA) is inserted between the two FILE INSERT (.FI) commands. Now the final FILE INSERT (.FI) command and a RETURN, and the program is finished.

Not only is the program finished, but as soon as you go back and add the rest of your information into your data file, the system will be completed. Don't forget that your data file is a NON-DOCUMENT file.

After having gone to all of this work, I found that one more program — actually two programs — would add a nice touch for me. I depend heavily on my desktop

Continued on Page 50

Getting Started with . . .

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

Microsoft Word

Well, it is here at last; at least in beta version. For the last few days, I've had a chance to try out a beta version of Word 5.0. By the time you read this, the final version should be shipping.

Here are some observations about 5.0. Just remember that it is a beta version and I've had only a few days to get acquainted, so you will see a few more changes that I've missed or last minute changes by Microsoft.

Word will now use expanded memory and run under OS/2 (oh, goodie). Expanded memory use will increase speed and capacity for such memory intensive operations as Sorting, and Search and Replace. Many times I've had to restart a search and replace operation because of memory limitations. Now all I need is some expanded memory.

were all in 43 line mode, which I later changed to 25 lines in the hope you would be better able to read them when they appear in REMark.

Figure 5 is a sample help screen. If you find you still want better help screens, contact Rinearson Support Associates, 219 First Avenue N., No. 410, Seattle, Washington 98109, telephone (206) 824-4132. They have a Microsoft Word Companion disk with more help screens. The disk for Word 4.0 was \$19.95, but I believe the cost has increased for version 5.0. In addition to the help screens, a lot of macros are included. They also have a disk of style sheets available.

You will have to redo your 4.0 macros. A conversion macro is furnished that will do most of the grunt work (they say). Not all the macros furnished with 5.0 are

they have finally added is a table showing some of the program limits, like the maximum document size is 8.3 MB. One limit still left out is the maximum size of a printer driver. I've heard it is around 64 KB. Mergeprd gives a warning around that size, but I've loaded PRD files of 64 KB and Word does not complain.

For all you laser printer users, be aware that the printer driver file format changed again! If you have customized your printer drivers (the .prd files), there may be some unpleasant surprises for you. Microsoft explains how to convert 4.0 drivers to the new format using mergeprd.exe. Unfortunately, that procedure does not always work, as I have discovered. I have created my drivers with Laser Fonts® by Softcraft. To date, I've been unable to get one to work properly after conversion. Everything seems to work, but the downloaded fonts will not print. A Softcraft technician stated they have a beta version of Laser Fonts for 5.0 which they promised to send "real soon now".

In the interim I've had to create drivers based on the ones furnished with 5.0 and generate new font files to work with them. That has eliminated my using special symbol sets and fonts for a while. Oh, you can now tell Word that you really use 8.5 inch paper in your LaserJet and have been lying all these years about the 8 inch stuff. The driver now knows all about the "unprintable" area.

The ability to switch between a text screen and a graphic screen was added with 4.0. Showing either 25 or 43 lines on screen was an option only when starting the program. Now no longer do you have to exit from the program to change the number of lines displayed on screen. That is an option on the Option menu, Figure 2. The last two modes used are remembered and toggled between with ALT-F9. The Options menu now includes what was under the Window Options menu and takes up over half the screen. The "printer display" selection is gone and replaced with "show layout" and "show linebreaks". All the color options offered for the window background, menu, borders, etc. are set from this menu. The cryptic "visible" field on the old Options menu is now called "show non-printing symbols".

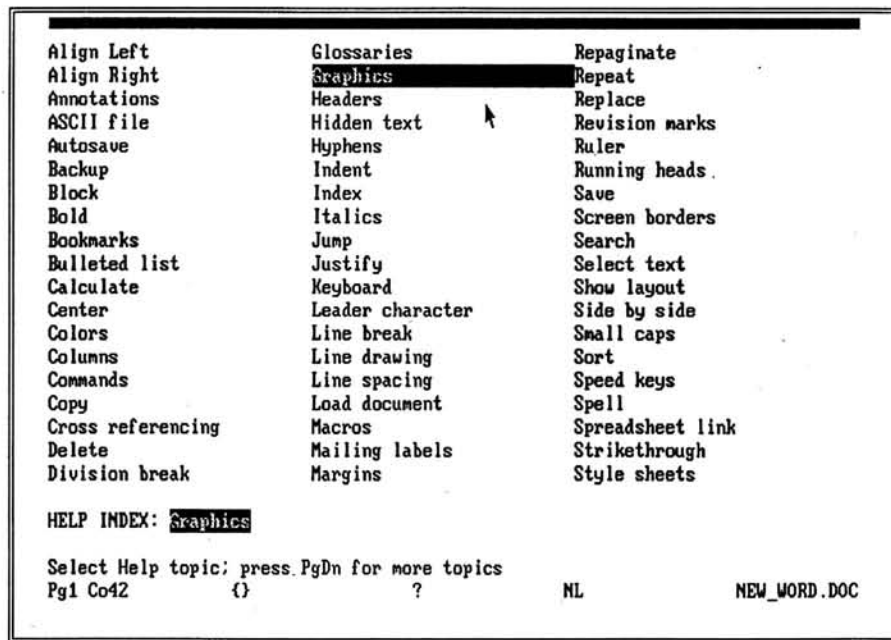


Figure 1
Help Index

The help screens have been improved and style sheets are better explained in the manual. As you can see from Figure 1, the number of topics in the Help index has been increased from the 29 in Word 4.0 to 93 entries. Well, trust me, there are 93 entries. The others show on the next screen unless you use a 43 line display. My initial screen captures

bullet proof. There is a nice one to copy your Word file and all its supporting graphic files to another drive; however, it will fail if you are in "show layout" mode.

I will not comment too much on the documentation; being the 50th generation photocopy, it reminded me of a Unix manual. You should see Microsoft's usual slick manual in the final version. One item

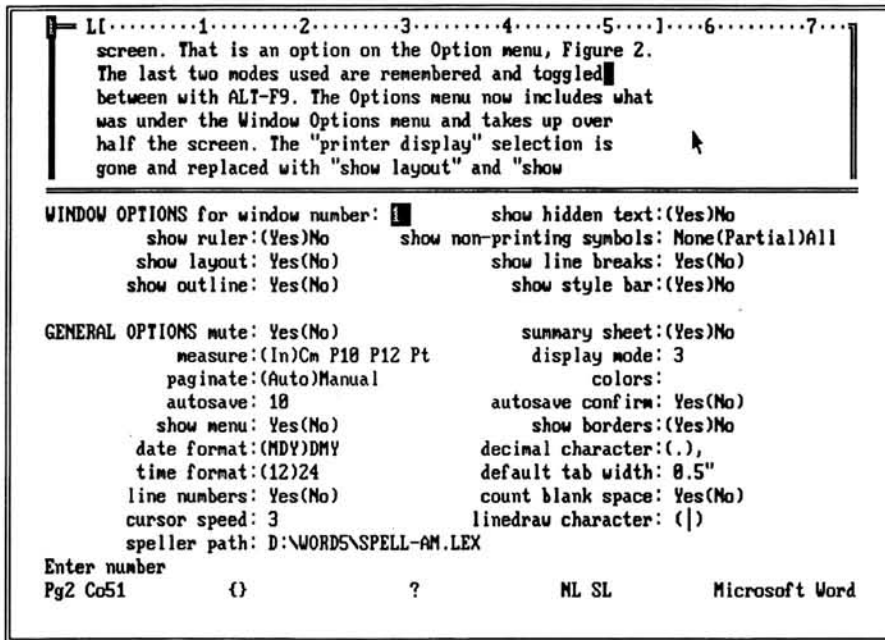


Figure 2
Options Menu

Automatic pagination is available for the first time. I really never missed it, but now that it is here it is quite handy. A word count is still not available without printing or spelling the document. These two word counts still do not agree, but they are close enough for my use. I had to discard the word count program that came with version 3 as it locked up everything when used with version 4.

Another selection on the Options menu is automatic saving of your files. This is more than just a regular Transfer Save on a timed basis; a whole new set of files are created. You can specify the time delay, in minutes, between saves (or turn it off). I know one person that will really appreciate this feature. He got so engrossed in a project recently he did not realize he had not saved until two hours of work fell into the bit bucket when his machine locked up.

One more related new command is Transfer Allsave. Very handy, particularly if you work with multiple documents in windows as it saves all open documents, style sheets and the glossary with just one command. The Transfer Save command now has a choice for text-only to produce an ASCII file without having to change to the plain print driver, adjust margins and then reset everything.

File handling is greatly improved. For example, the Transfer Load command now shows (Figure 3) the parent directory, subdirectories and all drives on screen in addition to the .DOC files.

Navigating around on a hard disk to find a file is much easier as you no longer have to keep changing the Transfer Options entry to search another directory. A CTRL-F7 will display all the DOC files as before. In addition, all available drives are

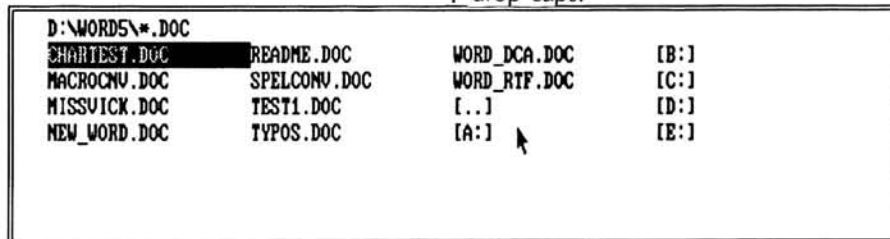


Figure 3
Transfer Load Display

shown as [A:] or [D:], the parent directory is [..] and a subdirectory is shown as [RE-PORTS]. Just click on the drives and directories, a la Windows, to arrive at your destination. The feature is available on all

file selection modes, including linking graphics and attaching style sheets.

The speller, Figure 4, now operates like the Thesaurus. No more giant wait for "Saving work file" and then, presto, one blank screen. The document still appears on screen and a small window is opened at the bottom to display incorrect words and choices. Plus there is an undo command. The mark option is no longer present; however, you can insert a special character by using the "correct" function.

If you turn "revision marks" on before starting spell, all the corrections will be marked in the document.

The biggest change to Word is the improvement in handling graphics. All you could include in 4.0 were printer files. Now several graphic file formats may be included in the document including TIFF, PCX/PCC and Lotus PIC files, as shown in Figure 5.

The graphic feature also allows easily doing some things with text that were awkward before. Things like full width headlines above multi-column text or drop caps.

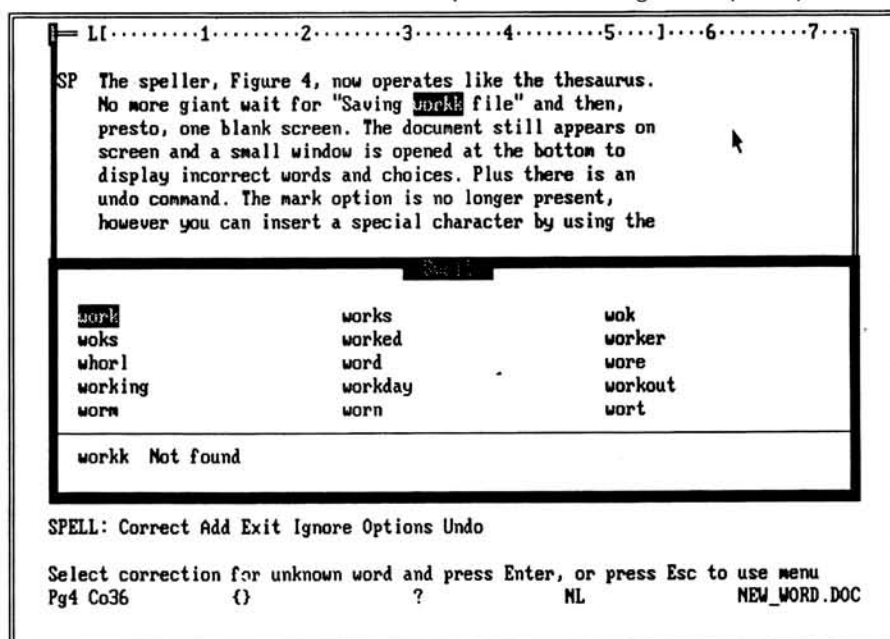


Figure 4
Speller

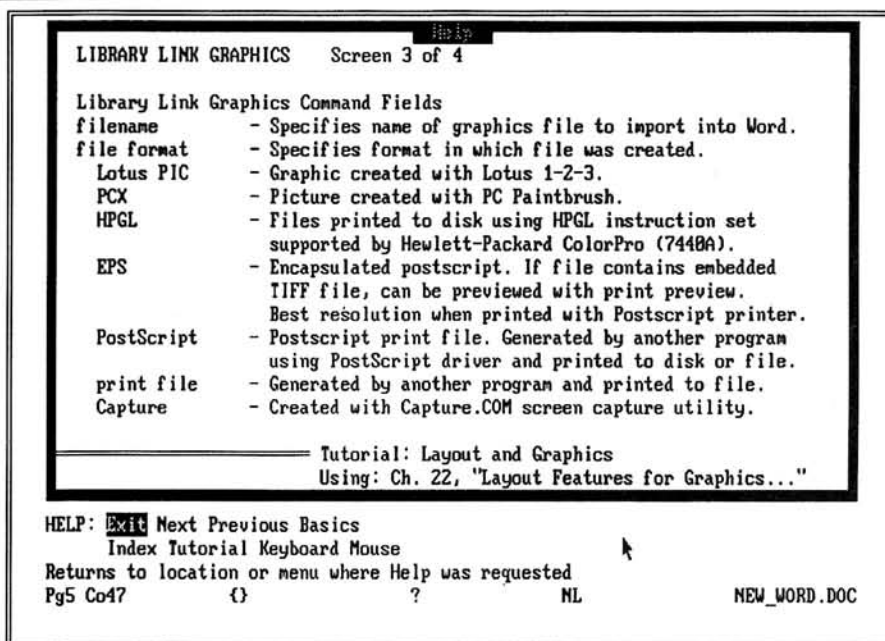


Figure 5
Compatible Graphic Formats

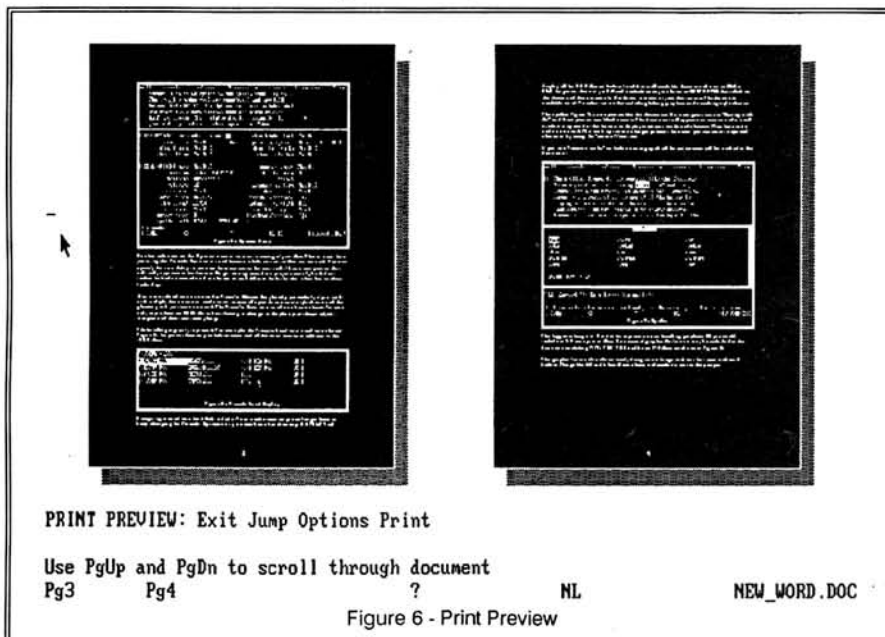


Figure 6
Print Preview

is a circle, text wrap will be around an imaginary rectangle containing the circle. Although the graphic can be sized, the bad news is, that is all you can do with it; no rotation, flipping or reversing is available.

All that will have to be done with a graphics program. With the Library Link Graphic command, the name and path to your graphic image is inserted into the document, including the necessary hidden text. Next, the Format pOsition command is used to size and position the image on the page. By setting the option "show layout", the rectangular frame around the image will show on screen and text will wrap around, allowing you to see

the effect of changes in real-time.

Included with Word is a screen capture program to provide graphics from your programs. Besides capturing the screen, it can rotate the image and crop it. This program was used to capture the Word screens you see here.

On the Print menus, you will find a new choice to change the graphics resolution and a preView function. Preview gives you a look at one or two pages as they will print, including graphics. Figure 6 is two pages of this article. The two pages may also be shown as facing pages, as in a book. Running heads, footnotes, page numbering and the like all show on screen in greeked format. Around 18

points in size text starts to become readable. This feature appears to be a subset of the Pageview program which was available free with 4.0. Unlike Pageview, no zoom feature is provided and no edits, even margin changes may be made. But then, you don't have to exit Word to use it, either.

If you look close at the left side of the preView screen, you will notice a small horizontal bar, like a hyphen. It appears to be the same little bar you see with the normal document on screen and shows you your relative position in the file. Clicking on the left side scrolls around in the document here, also.

A while back I had an occasion to want a document with odd pages having a wide right margin and even pages with a wide left margin. Now, the reverse is very easy to do with gutter margins. I struggled for some time and finally gave up and modified my layout. Later, after the project was finished, I discovered a method using one division format for odd pages and another for even. I think it would have worked, but lo and behold, 5.0 includes a "mirror" format to do just what I was trying to do. Maybe they do listen after all!

Some other improvements I have noted include:

- A selection for running heads to start at the paper edge (the standard prior to 5.0) or at the left margin. Now if you change margins, it is not necessary to redo all the running heads.
- Default command for Library Run is "command", to load command.com as a shell. This allows you to run several commands before returning to Word without having to remember to type "command". You might want to change your prompt to something like "Type EXIT" to remind you you are in a shell. See Figure 7 for a suggestion to accomplish this in a batch file.
- During installation, setup allows you to preselect some of the default settings, such as showing or not showing the menu and the default paper size.
- Side-by-side paragraphs and columns can be shown that way on screen and still be edited.
- Several changes in function key assignments. CTRL-F4 now switches text between upper- and lowercase. ALT-F4 toggles show layout. ALT-F7 is show line breaks and CTRL-F9 is print preview.
- Markings on the ruler will correspond to either 10 pitch or 12 pitch character widths, depending upon the setting for "measure" in the Options menu.

Yet another feature new to Word is one called "bookmarks". Besides the obvious use to return to a selected spot in a document, they can be used for cross-referencing. For example, you might have an entry on page 25 where you refer readers

```

:batch file to call Word and change prompt
cd \word\data
prompt Type EXIT $p$g
word
prompt $p$g
cd \

```

Figure 7

to an earlier page. By using the bookmark feature your reference to the previous page will be updated if subsequent edits change the page number. Besides pages, you can refer to illustrations, tables, or paragraphs that are automatically numbered.

So where does Word stand with Word Perfect? In my opinion, they are in a dead heat. Word still has much better control over the typography. Word Perfect's use of font size terms like "Big" and "Tiny" leave me wondering why anyone bothered to think up the point system for

measuring type size. Word Perfect does appear to have the edge on handling graphic images because you can do more than just size and place the picture.

The changes have done a lot to improve Word; it has strength in the area I need, which is good control over the printed product.

```

:batch file to call Word and change prompt
cd \word\data
prompt Type EXIT $p$g
word
prompt $p$g
cd \

```



Continued from Page 46

Figure 12
The program to print all of the data by generic name

```

..This file is called CHECKDAT.PGM to print CHURCH.DAT <
..This is to print all of the data by its generic name to <
..aid in checking the data for accuracy <
..Typed (date) <
.PL 11 <
.MT 0 <
.MB 0 <
.PO 0 <
.DF CHURCH.DAT M
.RV SORTNAME, ADDSTREET M
.RV CHURCH, ADDMAIL, CITY, STATE, ZIP, OFFICEPH M
.RV LASTNAME, FIRSTNAME, POSITION, TITLE, HOMEPH M
CHURCH: &CHURCH& <
MAILING ADD: &ADDMAIL& <
STREET ADD: &ADDSTREET& <
CITY/ST/ZIP: &CITY&, &STATE& &ZIP& <
OFFICE PHONE: &OFFICEPH& <
POSITION: &POSITION& <
TITLE: &TITLE& <
FIRST NAME: &FIRSTNAME& <
LAST NAME: &LASTNAME& <
HOME PHONE: &HOMEPH& <
.PA <
-----P

```

rotary card file for lots of information; especially phone numbers. In about 10 minutes total, I copied, edited, tested, adjusted ("fine tuned"), and retested these two programs to print these addresses

and phone numbers on my 2 X 4 inch rotary file cards. The first program prints the roster information and the second program prints the phone listing. This way I can look up a phone number by the

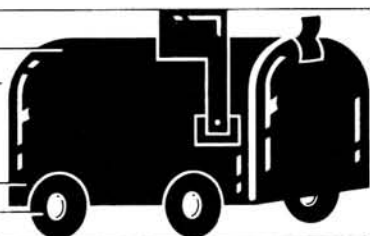
church's name or the pastor's name. Figure 11 is the print program to print the addresses to the 2 X 4 inch rotary file cards. The program for printing the phone listing, named INDEXPH.PGM, has the same changes in the print format that are in the phone list for the roster in Figure 7.

I have written one more print program that was not a part of the original system. I wrote it when I was trying to check my date entry in another system with almost 100 data entries. This program is called CHECKDAT.PGM and is printed in Figure 12. I have found it to be very helpful to me. You may find it is helpful to you also.

If you have followed this article to this point without giving up, and if you typed these or similar programs and files, you will be able to construct any other data file and print program systems you may need. Keep it as simple as possible to get the job done, but do keep good records and printed copies of your programs. Most of all, remember that all of this work is to do a job, not just to spend time at the computer. It's time for the computer to work for you.



MOVING?



Don't Miss A Single Issue!
Let us know 3-4* weeks before you move!

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



Z-100

Paul F. Herman
3620 Amazon Drive
New Port Richey, FL 34655

SURVIVAL KIT

The Technical Information Gap

Quite a bit of the mail coming in as a result of this column is from Z-100 users who crave in-depth technical information. I can appreciate that, because I, too, am interested in knowing everything I can about the '100. And in recent times, there doesn't seem to be many places you can turn to for such information . . . unless you're lucky enough to have an active users' group in your neighborhood.

As far as orphan computers go, the Z-100 is pretty well documented. Zenith published a decent technical manual for the machine, and the service literature and spec sheets describe its operation with a fair amount of detail. Of course, if you're not a Zenith Service Center, you don't have access to all the service literature, but most of the info you need to program the machine is in the Z-100 Technical Manual (a two volume set).

Although most of the information about the Z-100 hardware is available, portions of it can't be easily understood by the novice. One of my goals in writing this column is to try to bridge the gap between raw technical information and practical applications. After all . . . having the Motorola spec sheet for the MC2661 Enhanced Programmable Communications Interface in front of you is one thing . . . but knowing what to do with it is another.

I'd enjoy writing technical stuff every column, but I'm afraid the average person would quickly become bored with that. So you techies are going to have to be patient — we'll bite off a chunk at a time. You're in luck this month though, because I'm going to talk about programming the Z-100 keyboard chip. If you're

just a casual programmer, don't be scared away. I'll try to keep it light, and throw in a few code examples in BASIC.

How the Keyboard Hardware Works

The Z-100 keyboard circuitry is a model of simplicity, or complexity, depending on how you look at it. In addition to the array of key switches, one might expect to find a bundle of individual gates or encoder chips to convert the key switch positions to key codes. But instead, the keyboard circuit is composed mainly of only four chips. The secret to this low chip count is that the main keyboard encoder chip is a full-fledged 8-bit microprocessor, complete with two I/O ports, 1000 bytes of ROM, 64 bytes of RAM, internal clock and timer. The Intel 8741A Universal Interface Microcomputer used as a keyboard encoder is really a little computer that runs independently inside your Z-100. It has its own ROM program which handles key code assignments, autorepeat, and key buffering. There are several other chips associated with the keyboard circuit, but they are used only to generate the key click, bell, and reset signals.

Fanatical hackers might also want to note that the program in the 8741A chip is stored on a UV erasable EPROM, so you could actually modify the keyboard encoder programming if you want — the programming instruction set is given in the Intel spec sheet for the 8741A. I can't even imagine why you would want to re-program the keyboard encoder, but given the possibility, someone will think of a reason.

After seeing that the keyboard encoder is really a computer in itself, it's

easy to explain how the keyboard works. The 8741A scans the keyboard switch array by reading its two I/O ports (one port scans the rows, the other scans the columns). When a switch closure occurs (or a switch opening in up/down mode) the key switch is identified, and an appropriate key code is placed on the data bus. The RAM in the encoder is used to provide a 17 key first-in, first-out buffer in case the processor can't service the keyboard right away.

Programmable Features of The Keyboard Encoder

The keyboard encoder can be programmed to operate in one of two different modes. The default (power up) mode is called the ASCII scan mode. In ASCII mode, the keyboard matrix is scanned until a key is pressed. After the key switch is read, scanning resumes to see if another key will be pressed. Two keys, at most, are read at a time and decoded (thus allowing key modifiers such as SHIFT or CTRL to be used). When one of the keys are released, scanning resumes again. The key codes read at the data port during ASCII scan mode are the ASCII codes you are familiar with. Special codes are generated for keys that don't have an ASCII equivalent (like the function keys).

The other mode of keyboard operation is called UP/DOWN (or event driven) scan mode. In this mode, the encoder scans continuously, regardless of what, or how many keys are down. In addition to generating a keycode when the key is pressed, a unique keycode is also generated when the key is released. The key codes read from the data port in UP/DOWN mode are arbitrary codes, and not

the standard ASCII key codes. Each key has a unique up and down code.

The ability to switch between normal ASCII mode, and UP/DOWN mode, is one of the nice things about having a microprocessor for a keyboard encoder. Instead of requiring a totally different keyboard circuit, all that is necessary is for the internal encoder program to use a different scanning algorithm.

In addition to the two scan mode choices described above, you may also select how keystrokes will be detected by your program. The way it is normally done (by the BIOS) is by using a hardware interrupt generated by the keyboard encoder whenever a key code is available at the data port. This interrupt is serviced by an interrupt routine which reads the key-code, and places it in the type-ahead-buffer. Alternatively, you may tell the keyboard encoder not to generate interrupts, in which case you will need to continuously poll the keyboard status to see when a key is struck. More on this later.

Another feature that can be enabled or disabled by software is the key autorepeat. When enabled (the default), this feature will cause the key to automatically begin repeating after it is held down for a short time. The repeat rate (11 keys per second) cannot be changed by software, although the FAST REPEAT key can be used to jump it to 28 keys per second. The FAST REPEAT key will cause the key to repeat **even if autorepeat is disabled**.

We all know that the Z-100 keyboard makes a little click sound when the keys are pressed (and released, in UP/DOWN mode). This feature can also be turned off with software. The key click is generated by the same circuit and oscillator as the bell sound. Both are 1 kHz tones, but the short 10 ms duration of the key click makes it sound like a 'click'.

Lastly, your program can disable the keyboard encoder completely, so that all keystrokes will be ignored. After disabling the keyboard, the only way to get it back again is to send an 'enable keyboard' command, or do a master reset (CTRL RESET).

For further information about the operation of the keyboard encoder, you might want to refer to the Z-100 Technical Manual. A fairly detailed description is given on theory of operation and programming of the encoder.

Let the Programming Begin!

'Nuff talk. I think we're ready to do something with the keyboard encoder. There are three ports that are used to communicate with the encoder chip;

Data Port	0F4H	...	24
4 decimal			
Command Port	0F5H	...	24
5 decimal			
Status Port	0F5H	...	24
5 decimal			

BASIC Program to Experiment with Keyboard Encoder Commands

```
100 REPEAT=0 : GOTO 140
110 PRINT"Hit 'R' to toggle AutoRepeat feature..."
130 IF INPUT$(1)<>"R" THEN 110
140 IF(INP(&HF5) AND 2)=2 GOTO 140
150 IF REPEAT THEN REPEAT=0 : OUT &HF5,2 ELSE REPEAT=1 : OUT &HF5,1
160 GOTO 110
```

Listing 1

The data port (at 0F4H) is a read-only port. (What would you expect a keyboard to do with data you wrote to it, anyway?) The command port (at 0F5H) is used to tell the keyboard encoder chip what you want it to do. And the status port (also at 0F5H) is used to see if the keyboard processor is ready, or if there is a keycode available on the data bus. Obviously, the command port is a write-only port, and the status port is a read-only port. (Otherwise, how could they be at the same address?)

The most obvious thing we might want a program to do with the keyboard encoder is read keystrokes, but let's leave that for a minute. It's easier to demonstrate how to write commands to the keyboard encoder by writing a program that does some other things. How about a sample program that enables or disables the autorepeat feature. (We could also do a simple program that turns the key click on or off, but I think that subject has been run thoroughly into the ground in every Heath/Zenith publication I have ever read.)

Consider the simple BASIC program shown in Listing 1. Line 100 assures that the keyboard encoder starts out in autorepeat mode. Line 130 waits for you to hit a

key. If the key is a capital 'R', the autorepeat mode is toggled on or off, otherwise, the prompt message is re-displayed. Line 140 is a loop that checks to make sure the keyboard processor is ready to accept a command. Actually, at the speed which GW-BASIC executes commands, there isn't any way the program could keep up with the keyboard processor, so this line could be eliminated from the listing without any ill effects. But it is included here to demonstrate good programming practice. If you program in assembly language or a compiled language, you're looking for trouble if you don't poll the keyboard processor status before issuing commands. Line 150 of our program is where the keyboard encoder receives its orders.

Every time you want to send a command to the keyboard encoder, you should do two things;

1. Check bit 1 of the status register to make sure the keyboard processor is ready to accept a command. If bit 1 is zero, it's okay to proceed.
2. Write the command to the command port.

Want a simple subroutine that can be used to issue any command to the keyboard controller? Listing 2 gives versions

Routines to Issue a Command to the Keyboard Encoder

BASIC Routine (Command Byte is in Variable C)

```
1000 IF(INP(&HF5) AND 2)=2 GOTO 1000
1010 OUT &HF5,C : RETURN
```

Listing 2a

'C' Language Routine (Command Byte is Passed as an Argument)

```
keycom(c)
int c;
{
    while(inp(0xF5) & 2);
    outp(0xF5, c);
}
```

Listing 2b

Assembly Language Routine (Command Byte is Passed in Register DL)

```
KEYCOM PROC NEAR
IN AL, 0F5H
TEST AL, 2
JNZ KEYCOM
MOV AL, DL
OUT 0F5H, AL
RET
KEYCOM ENDP
```

Listing 2c

of such a routine in BASIC, 'C', and assembly language.

By now you must be getting curious about what types of commands are available, other than the ones to turn autorepeat on and off. Table 1 gives a complete list of the commands that can be sent to the keyboard encoder.

You can try experimenting with some of these commands using your own variation of the BASIC program presented above. Try modifying the program to turn the key click on/off, or enable/disable the keyboard. May I strongly suggest that you save your first trial of the keyboard disable/enable program on disk before trying it, because if you did something wrong you may have to reboot to regain control.

Experimenting with the Scan Modes

Now that we're all experts in sending commands to the keyboard encoder, it's time to move on to more urgent matters. I know that turning the key click on and off is a pretty impressive maneuver, but we generally expect more from a well rounded keyboard interface program. In particular, it would be nice if it would tell us what keys are being pressed.

As mentioned above, the Z-100 offers two different methods of scanning the keyboard, and we're going to try both of them. The sample program shown in Listing 3 serves two purposes; it demonstrates how to use either mode with a BASIC program, and it also tells you what key codes are being generated (for those of you who don't have access to the key code tables in the tech manual).

Some explanations are in order . . .

After you select which scan mode you would like in line 100, the program writes a command to the keyboard which disables the hardware interrupts generated by the encoder. If we don't do this, then nothing will happen in our BASIC program, because BASIC's keyboard interrupt routine will steal all the keystrokes before we get 'em. Even after disabling the hardware interrupt, you'll still notice that a character comes up missing now and then (in other words, you hit a key and no key code number is displayed). This indicates that BASIC is still lurking in the background somewhere, watching over things. Further evidence of this 'big brother' feature of BASIC is that if you continue to hit Control-C, you can regain control (when using ASCII scan mode). As an interesting aside, when you are in UP/DOWN mode, you can sometimes cause the program to break by hitting the 'G' key, since the down code for 'G' is an ASCII three, which is the same as Control-C. Of course, once you break out of the program, you are stuck in UP/DOWN mode, which is sure to cause unexpected results, followed quickly by a crash. Moral of the story . . . be sure to save the sample program before trying UP/DOWN mode.

Keyboard Encoder Command Codes

COMMAND	CODE (in Hex)
Reset	00
AutoRepeat ON	01
AutoRepeat OFF	02
Key Click ON	03
Key Click OFF	04
Clear FIFO buffer	05
Generate Key Click	06
Generate Bell	07
Enable Keyboard	08
Disable Keyboard	09
Event Driven (UP/DOWN) Mode	0A
ASCII Scan Mode	0B
Enable Interrupts	0C
Disable Interrupts	0D

Table 1

Back to the program . . . line 120 is where we select the proper command; 0AH for UP/DOWN mode, or 0BH for ASCII scan mode. Line 150 is a loop similar to the one we used to make sure the keyboard processor was ready to accept a command. Except this loop is checking to see if a key code is ready at the data port. If bit 0 of the status register is set (1), then a key code is ready, otherwise, keep trying. This method of reading the keyboard chip is referred to as 'polling'.

Line 160 is responsible for reading the key code from the data port, and displaying it on the screen. And the code at line 500 is the little subroutine from Listing 2 that sends a command to the keyboard encoder.

This sample program is really rough, but I wanted to keep it as simple as possible, and still demonstrate how to access either scan mode. Obviously, if you are going to write a program that uses UP/DOWN mode, you would want to make sure that the keyboard is left in ASCII scan mode before you exit. Otherwise, DOS wouldn't have a chance.

You may have noticed that I haven't mentioned anything about 'alternate keypad mode' or 'key expansion'. These special keyboard functions are implemented by the Z-100 firmware — not by the keyboard encoder. Whenever you read the keyboard encoder directly (through polling, or an interrupt routine) all you get is the raw key scan codes.

What Use is All This?

A good question! Almost every program you write will probably be quite content to use the normal language keyboard interface functions. In BASIC, the INPUT, INPUT\$, and INKEY\$ commands will handle just about any situation I can think of. GW-BASIC even has the ability to implement event-trapping subroutines via its ON KEY command.

Pascal and 'C' also have a fairly nice set of keyboard input functions, as long as you can be content with using the normal ASCII scan mode. If you are an assembly language programmer, MS-DOS function calls offer a good keyboard interface, but again, you are limited to the ASCII mode.

As far as I can see, there are only a few legitimate reasons for going to the trouble of accessing the keyboard encoder directly;

1. A memory resident routine may want to avoid using any DOS function calls.
2. You may want to prevent any other programs from intercepting your program's keyboard input.
3. You may have a real-time application that requires use of the UP/DOWN mode.

Reason number one is probably not a valid excuse for going directly to the keyboard hardware. Even though DOS is not re-entrant, you can solve this problem by calling a BIOS keyboard input routine. Not only is this much simpler than writing an entire keyboard input routine yourself, but it also allows you to take advantage of the BIOS type-ahead-buffer.

Reason number two is valid, providing your program must maintain control. Accessing the keyboard hardware directly will prevent most (but not all) memory-resident utilities from working. If this is your intention, then have at it. Just keep in mind that this technique will generally cause your program to be labeled as 'ill-behaved'. (As opposed to 'well-behaved' programs, which follow all the rules, and only use DOS function calls).

Reason number three is the best excuse for writing your own keyboard input routine, and is probably the least understood of all. What do we mean by a real-time application? Well, the most common type of real-time programs are games, but any type of program that depends on knowing exact key movements

BASIC Program to Experiment with Keyboard Scan Modes

```

100 CLS:INPUT" ) ASCII Scan Mode 2) UP/DOWN Scan Mode : ",M$
110 IF M$<>"1" AND M$<>"2" THEN BEEP : GOTO 100
115 C=&HD : GOSUB 500
120 IF M$="1" THEN C=&HB ELSE C=&HA
130 GOSUB 500
150 IF (INP(&HF5) AND 1)=0 THEN 150
160 PRINT INP(&HF4) : GOTO 150
500 IF (INP(&HF5) AND 2)=2 THEN 130
510 OUT &HF5,C : RETURN

```

Listing 3

might be a candidate for UP/DOWN scan mode.

An Interrupt Driven UP/DOWN Mode Keyboard Input Routine

If you are writing a program that needs to use the UP/DOWN scan mode, the most versatile way of doing it would be to use the interrupt driven mode of operation, and write a simple interrupt routine that services the keyboard interrupts. Why not simply poll the keyboard? Well, the main reason is that this ties your program down to polling duties, when it could be doing something more productive. By using interrupts, the keyboard interrupt routine will automatically service any incoming keystrokes without any attention from your application program. Then when your program is ready for a key code, it can simply pull one out of the key buffer.

The assembly language program shown in Listing 4 gives a working, practical example of how an interrupt driven keyboard routine might be implemented. This program uses the UP/DOWN mode, but it can be easily converted to use the ASCII scan mode by simply changing the key code equates, and omitting the code that places the keyboard encoder into the UP/DOWN mode.

To assemble this listing into an .EXE program, you'll need the Microsoft MASM assembler, and the LINK object module linker. Create a source file named UD-TEST.ASM using a text editor, and then issue the following commands;

```
MASM UDTEST;
LINK UDTEST;
```

Here's a brief explanation of how the program works. The main line program begins at the START label. This is the entry point when the program is invoked from DOS. The program first saves the original keyboard interrupt vector which points to the BIOS interrupt routine. We need this vector so that interrupts not generated by the keyboard can be passed to the original routine, and we also need to save the vector so it can be restored when we end our program. Next, we install our own keyboard interrupt routine (KBD__INT) in the interrupt table. The keyboard encoder is placed into the UP/DOWN mode, and finally, we call the main program routine (TESTIT).

When the TESTIT routine returns (ESC is entered at the keyboard), we need to clean up before we quit. The keyboard encoder is put back in its default ASCII scan mode, and the original BIOS keyboard interrupt vector is restored to the interrupt table. Then we return control to DOS.

The TESTIT routine allows us to move a 'dot' around on the screen by pressing the arrow keys. This is a very simple routine that essentially does nothing, except provide us with a demonstration. In fact,

Listing 4

```
; UDTEST: This program demonstrates how to use UP/DOWN scan mode in an
; assembly language program. It sets up its own keyboard interrupt routine,
; switches the keyboard to UP/DOWN mode, and then allows you to move an
; asterisk (dot) around on the screen using the arrow keys.

INT_KD          equ     46H          ; keyboard/vertical sync interrupt
pt
ESC_K           equ     4FH          ; ESC key down code
UP_K            equ     3BH          ; UP arrow key down code
DOWN_K          equ     3AH          ; DOWN arrow key down code
RIGHT_K         equ     33H          ; RIGHT arrow key down code
LEFT_K          equ     3FH          ; LEFT arrow key down code

STKSEG segment stack
db 10 dup(?)    ; don't need much stack
STKSEG ends

DATSEG segment
KEYCODE db 0          ; current key code
CLS db 27,'x5',27,'E' ; initialize screen
db 'Use arrow keys to move dot, or ESC to end...'
db 27,'Y',43,71,'*',27,'D$'
CURSON db 27,'y5$'    ; turn cursor back on
UP db ' ',8,27,'A*',27,'D$' ; move dot up
DOWN db ' ',8,27,'B*',27,'D$' ; move dot down
RIGHT db ' ',*,27,'D$' ; move dot right
LEFT db ' ',8,8,'*',27,'D$' ; move dot left
DATSEG ends

PGMSEG segment
assume cs:PGMSEG, ds:DATSEG, ss:STKSEG, es:nothing

START: mov ax, DATSEG ; get our data segment
mov ds, ax
mov al, INT_KD ; save system keyboard interrupt
mov ah, 35H
int 21H
mov cs:BIOS_I, bx
mov cs:BIOS_I+2, es
push ds ; now, install our interrupt routine
mov dx, offset KBD_INT
mov ax, PGMSEG
mov ds, ax
mov al, INT_KD
mov ah, 25H
int 21H
pop ds

ST1: in al, 0F5H ; put keyboard in UP/DOWN mode
test al, 2
jnz ST1
mov al, 0AH
out 0F5H, al
call TESTIT ; call the main program

ST2: in al, 0F5H ; put keyboard in ASCII scan mode
test al, 2
jnz ST2
mov al, 0BH
out 0F5H, al
mov dx, cs:BIOS_I ; restore system interrupt vector
mov ds, cs:BIOS_I+2
mov al, INT_KD
mov ah, 25H
int 21H
mov ah, 4CH ; return to DOS showing no errors
mov al, 0
int 21H

TESTIT PROC near
mov dx, offset CLS ; clear screen, turn cursor off
mov ah, 9 ; and put '*' in center
int 21H
TEST1: mov al, KEYCODE ; get the current key code command
mov dx, offset UP
cmp al, UP_K ; UP arrow key?
je TEST2
mov dx, offset DOWN
```

```

    cmp     al, DOWN_K           ; DOWN arrow key?
    je      TEST2               ;
    mov     dx, offset RIGHT    ;
    cmp     al, RIGHT_K         ; RIGHT arrow key?
    je      TEST2               ;
    mov     dx, offset LEFT     ;
    cmp     al, LEFT_K          ; LEFT arrow key?
    jne     TEST4               ; if none, skip it
TEST2:  mov     ah, 9            ;
    int     21H                 ; move the dot
    mov     cx, 4000H           ; slow things down a bit
TEST3:  loop    TEST3           ;
TEST4:  cmp     cs:KEYBUFF, 0    ; wait for another key code from
    je      TEST1               ; the keyboard interrupt routine
    mov     al, cs:KEYBUFF      ; get the keycode
    mov     cs:KEYBUFF, 0      ; reset to show no keycode available
    cmp     al, ESC_K          ; is this the ESC key?
    je      CLEANUP            ; yes, prepare to quit
    mov     KEYCODE, al        ; otherwise, store as current code
    jmp     TEST1              ; process the key code
CLEANUP: mov     dx, offset CURSOR ; turn cursor back on
    mov     ah, 9              ;
    int     21H                ;
    ret                        ;
TESTIT endp

```

; KBD_INT is our keyboard interrupt routine

```

    assume    cs:PGMSEG
KBD_INT: push  ax                ;
    in        al, 0F5H          ; check keyboard status
    test     al, 1              ; if none available, must be
    je       JMPBIOS            ; vertical retrace or light pen
    in        al, 0F4H          ; get key code
    mov     cs:KEYBUFF, al      ; save it in buffer
    mov     al, 20H             ; tell interrupt controller that we
    out     0F2H, al            ; have serviced the interrupt
    pop     ax                  ;
    sti      ;
    iret     ; and return from interrupt

JMPBIOS: pop    ax              ; do a far jump to the BIOS interrupt
    jmp dword ptr cs:BIOS_I     ; routine so it process interrupt

BIOS_I  dw      0,0             ; BIOS interrupt routine address
KEYBUFF db      0              ; keycode buffer (small buffer, eh?)

PGMSEG ends
end      START

```

since this program doesn't do anything of philosophical significance, it would have been a lot easier to simply poll the keyboard instead of using an interrupt routine. (In effect, our program is simply polling the one byte key buffer, instead of the keyboard encoder's data port.) But the idea here is to demonstrate how to write an interrupt driven keyboard routine.

The real heart of our program is the short interrupt routine which begins at the label KBD_INT. This isn't really a part of our main program, but is a separate little piece of code that gets called everytime a key is struck. It is invoked automatically everytime a keyboard interrupt occurs. The first thing our interrupt routine needs to do is check the keyboard status to make sure a key code is available. You might ask why this is necessary, if the interrupt was caused by a key being struck. The answer is that the particular interrupt used by the keyboard is also used by the video display to signal a vertical retrace, and by the light pen.

If we check the keyboard status and find that a key is not ready, we can safely assume that the interrupt was not generated by the keyboard, and simply jump to the original BIOS interrupt routine. If a key code is ready, our routine reads it, and stores it in the key code buffer. A normal program might want to have a buffer that is slightly larger than one byte, in order to prevent keystrokes from being lost while the program is doing other things. This key code buffer is normally referred to as a type-ahead-buffer. Of course, if you want to implement a type-ahead-buffer, you'll need to write special routines that take care of storing and retrieving key codes from the buffer. That's one nice thing about using DOS, or the BIOS, for keyboard input — all this overhead stuff is taken care of for you.

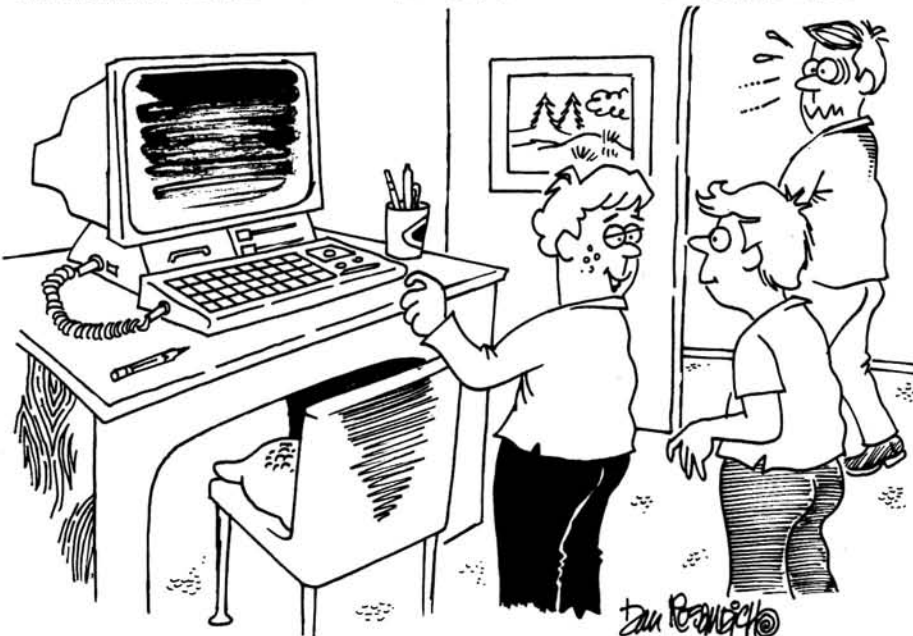
Once the key code is saved, the last thing we need to do is tell the interrupt controller we are done processing the interrupt. This is done by sending a 20H to port 0F2H. I don't want to get into a discussion of the 8259A interrupt controller chip here, so just take my word for it. If you don't do this, nothing will work. You might be asking, "how come we didn't do that before jumping back to the BIOS, when a key wasn't ready?" The answer ... the BIOS interrupt routine does it when it is done. You'll also notice that when we process the interrupt ourselves we must use an IRET instruction to return, but if no key code was ready, the BIOS interrupt routine will issue the IRET.

Wrapping It Up

In the last installment of "Z-100 Survival Kit", I included a Question & Answer section. I plan to include Q&A most of the time, but there just isn't going to be room this month. I promise to do better next time. Until then ... be sure to keep in touch!



"THIS OUGHTA BE GOOD... I JUST SWITCHED CLUB ZI'S MENUS WITH McDONALD'S!"





Some People Are Still Searching (and Searching) for a Better Way to Upgrade Their Systems than First Capitol Computer's Full System Upgrades...

send it to us to minimize the lost time, and we'll make every effort to turn the machine around within 48 ours of receipt.

Still Searching for Impossible Dreams?

We can wait, but pricing and availability on the best upgrade for you might not! Offers subject to change without notice! Better to lock your upgrade in now, while there's still time! *Prices listed below are based upon models available at the time this ad was placed. Call for current prices, availability and full details.*

Upgrades Currently Available

-Zenith Data Systems Upgrades-

FCC-150/286-UP: Z-151/2/8/9 or Z-161 to Z-286 80286 AT compatible (16 bit). \$1295.

FCC-150/248-UP: Z-151/2/8/9 or Z-161 to Z-248/12 12MHZ 80286 AT compatible. \$1895.

FCC-150/386-UP: Z-151/2/8/9 or Z-161 to Z-386 16MHZ 80386 computer. \$2495.

FCC-150/38625: Z-151/2/8/9 or Z-161 to Z-386/25 25MHZ 80386 computer. \$4795.

FCC-150/38633: Z-151/2/8/9 or Z-161 to Z-386/33 33MHZ 80386 computer. \$5695.

FCC-248/386-UP: Z-241/248 to Z-386/16 16MHZ model (32 bit processor). \$2095.

FCC-248/38625: Z-241/248 to Z-386/25 25MHZ model (32 bit processor). \$3995.

FCC-248/38633: Z-241/248 to Z-386/33 33MHZ model (32 bit processor). \$4895.

-First Capitol Clone Upgrades-

FCC-150/CL-AT: Z151/2/8/9 or Z-161 to First Capitol 12MHZ CL-AT (16 bit) Clone. \$995.

FCC-150/CL3825: Z151/2/8/9 or Z-161 to First Capitol 25MHZCL-386/25 (32bit)Tower. \$2295.

FCC-248/CL3825: Z-241/248 to First Capitol 25MHZ CL-386/25 (32 bit) Tower. \$1795.

These Also Work on Kit Models!

Add 2% shipping and handling (plus 6.725% sales tax, if a Missouri resident).



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

What Could Be Better than a Full System Upgrade?

Full System Upgrades aren't mere circuit boards, modifications, or kits! We take your original drives (and other components) and move them into a new chassis. The result is an almost entirely new machine indistinguishable from a unit right off the production line! No compatibilty problems, & all work is performed by factory authorized service technicians.

The Cost? A Small Portion of What a New Computer Would Cost You!

That's right! Since First Capitol uses parts from your old machine in the new cabinet, you only pay for what you need - and you receive a built-in trade-in credit for your old machine! We'll even provide trade-in credit for any third-party products which won't work in the faster machines towards replacements - and install them for you at no extra charge!

So, What's the Catch?

The catch is that since these upgrades involve components from your old computer, you'll lose use of that machine for about a week (counting shipping time in both directions). However, we will schedule your system upgrade before you

Graphics Printer or Epson FX?

Part 1

Choosing a Lightweight Printer, and What Characters You Get

When I bought my ZW-148, there wasn't much cash left over for the printer. This must be a common problem with amateurs. I decided to go for a simple 9-pin dot matrix printer with the mechanical features I wanted, and program the logic I needed. For this, I was helped by four years at work looking after a mixed bag of now nearly a hundred printers, of differing makes and models. I'm satisfied with the results; I can get up to about half the quality of a laser printer, and I can do anything the big office printers do (although they do it a helluva lot faster). I hope my experience may interest REMark readers. These articles, based on the work I've been doing and also several inches of technical manuals on my shelf, should help you to get even more out of your small printer.

First of all, what we're talking about. The lightweight 9-pin printers have a printhead about the size of a walnut, and are fairly flat. Compare this with the 24-pin office printers, which have a much heavier head and correspondingly powerful drive motors. The big printers usually have safety interlocks to prevent you from running them with the covers open, whereas you can run a 9-pin printer without the covers - but keep your fingers out of the way!

Choosing your printer

If you've already got your printer, I'll try to show you how to get the best from it. If you haven't bought it yet, here are a few points to watch for. First of all, decide what size paper you want to use. A 10" platen will let you print office stationery in portrait format only. Ask yourself whether you also want to use landscape format of 11" x 14" stock (or both!) and choose

John A. Day
5 rue Sauer
77500 Chelles
FRANCES

your width accordingly. Don't forget that an awkward document can be printed more easily on 11" x 14" and then reduced to standard format on a good photocopier; the result is much crisper than with a 100% copy ratio. 132 column continuous forms run to 15" or 16" wide; my 16-1/2" platen will take almost anything, including European DIN A3 paper in landscape position. You can get more on a line by using narrow, condensed characters; check the printer specifications and your eyesight.

For continuous forms the tractors can either push or pull. If they are above the platen and pull the paper away from the print area, buy a bigger wastebasket at the same time; every time you detach a document, you will leave one unused sheet in the tractors. Also, this sort of tractor may not let you reverse-feed the paper. Pushers let you tear off at the last sheet, and give nearly as good control of the forms. If you use cut sheets, look for automatic loading; you drop the sheet between the paper guides and press "Top Of Form" and the printer moves the paper to the print position. The bail bar must open to accept the paper, and snap shut at the right moment to feed the tip of the sheet past the top cover. Best loading is obtained if the printhead is near the middle of the paper to give extra guiding, but this means a separate mechanism for the bail bar. You will probably have to make do with a more simple device, where the printhead moves over the right stop to

nudge a lever linked to the bar and open it, then moves back into the print area to close the bar again. Check whether cut sheets can be loaded easily by hand; there are printers which can mix the paper up with the ribbon unless you have a sheet of bristol to poke into the paper path to help it around. For a lot of work with cut sheets, think about an automatic sheet feed (ASF) - but try it with your own forms before buying. This is particularly true with multi-part forms; we spent a lot of time once at the office discarding part 6 on all our invoice stock because our ASF baulked at more than five copies. Finally, if you switch often from continuous forms to cut sheet and back again, there are printers with a push-button change. They reverse-feed continuous forms to clear the platen and free the cut-sheet slot, and also disengage the tractors so that the forms can stay behind without advancing. However, this is mostly to be found on more expensive machines.

Now, look at character widths and styles. Every machine has the size I never use, 10 characters per inch (cpi). 12 cpi has a much more professional appearance. Look to see how many of the following pitches a printer can give: 20, 18 or 17, 12, 10, 9 or 8.5, 6, 5. For correspondence, you will need Near Letter Quality (NLQ); can the printer give it in all pitches? 17 or 20 cpi NLQ is not really necessary, but 12 cpi NLQ is. Can it print bold NLQ? The basic NLQ style is Courier or Prestige, which looks like ordinary typewriting. Your printer may accept cartridges or PCBs (printed circuit boards) with extra fonts; I always use Gothic instead of Prestige, as the sans-serif characters are crisper. Quadro is another sans-serif, a bit squarer than Gothic. Beware of

Anelia: it's a beautiful typeface, but I had to write my own printer drivers for my word processors to be able to use it.

Print wires - the pins that strike the paper - come in various thicknesses, from about 0.2mm up. If you intend on using NLQ fonts or high-resolution graphics, try to find a machine with wires between 0.2mm and 0.3mm thick. For mainly draft printing, fine print wires will give you a rather light image, and you will want something heavier. The difference in image quality between thick pins and thin pins is quite visible to the naked eye. To get an idea of pin size, look at a vertical draft line with a magnifying glass. If the dots touch each other, the pins are too thick for high-resolution work. At 0.33mm pin diameter, the dots nearly touch each other, but you can just make out a white space between them when the ribbon is new.

If you intend to develop software using a lot of special codes for the printer, a hex dump capability is useful. This is obtained by pushing a particular panel switch combination while turning the printer on; then, instead of printing in the usual way, the printer types the hex codes of everything it receives, including all control sequences. I've used this often for fast debugging, particularly with proprietary software when this was the only way to find out what the program was actually sending. And if you intend to print self-adhesive labels - check how easy it is to get at the paper path. If a label comes off the backing paper under the platen, it will wind around the pressure rollers and you may have to strip everything down to recover it. Never reverse-feed labels.

The best link to your computer is via a standard Centronics parallel connection. As well as data, this carries fault, off-line and paper-empty signals. You can also use a series interface, for example, if the parallel port is already occupied by something else; but you won't get full status information in this case. Printers can have both interfaces as standard, a series as extra, or a removable interface drawer where you can put one or the other.

Finally, look at ribbon cost. Unless you print thousands of pages, ribbons will dry out before they wear out because of the long loop of exposed ribbon across the page width. This is unavoidable, because the small drive motors on lightweight printers can't support the extra weight of a ribbon cartridge on the printhead. It does mean that ribbon life is measured in months, and not in millions of characters. Some printers have the cartridge at one end of the platen with a double loop of exposed ribbon; I would expect them to dry out twice as fast as the cartridges which reach the full platen width.

Every printer is reliable, according to the salesperson. I have my little list of

printers not to be used for seminars or training, but most seem ok if only one person uses them. When you've chosen the make you want, try and find somebody who's already got one and ask him about repairs.

How it works

All small matrix printers follow the same mechanical principles. The head moves across the paper at 15 to 20"/sec, printing on the fly. The platen only moves when the head is stopped; a stepper motor gives precise control. The solenoids which activate the print pins are rated for about 1000 strikes/second, corresponding to a 1/60" pitch on the paper. The current standard vertical spacing between pins is 1/72", but you may still find printers built to the old specification of 1/60". Whichever is fitted, the platen stepper motor will be geared so that three steps correspond to the distance between two adjacent pins - which puts the old step at 1/180", and the new at 1/216". The head needs about 3/8" to get up to printing speed, and another 3/8" to stop, so you can't print closer than this to the platen ends; this doesn't matter, as you always leave a small margin on any document.

Now is the time for some WARNINGS. Many printer instructions could drive the solenoids at up to 2000 strikes/second, which would rapidly damage them. Whenever you are using commands which don't go through the standard character generators, add program code to check specifically that you are not driving the printer beyond its ratings. Stay near the printer when you are using new or unfamiliar software; if the head stalls or whines, switch off immediately. Otherwise, you will burn out power transistors (it's happened several times at the office), and that means expensive workshop repairs. Manufacturers don't often quote the duty cycle, but you risk overheating at over 70% - this means to use 5 or 6 pins only if you want to score a line right across the page. If you need a heavy block firing all pins, like the ASCII character 219 (DB hex), then don't print on more than 70% of the line. Also, while we're on the subject of warnings, read the handbook on paper and head settings. If you have a manual setting for forms or cut sheets, remember that if it's set the wrong way, sheets will slip, and forms will jam after less than five minute's printing. The printhead *must* be set at the right distance from the paper. If it's too close, it will drag on the ribbon and you will end up with bald gearwheels on the ribbon drive mechanism (on one make of printer I use, a careless user can strip the gears in 15 minutes). Also, the head can catch on the edge of the paper if you print right up to it, which will offset further printing and you'll end up printing on the platen to the

right of the paper. A head too far from the platen is noisy and wears out fast. Remember to clean the platen periodically, particularly if you've been printing without paper (it does tear sometimes) or if you use NCR chemical copy sets - these last quickly coat the platen with an invisible slippery film. If the platen is greasy, cut sheets will slip, and an ASF will tend to feed paper skewed and tear it.

We have said that the maximum firing rate at full speed corresponds to one dot every 1/60". For standard pica, at 10 cpi, this gives us a matrix 6 dots wide for drawing characters. If we forget pin 9 for the moment, this means a 6 x 8 box. If you try sketching a few characters, you will find this is fine for squarish letters like E, F, or H; but is inadequate for diagonals or rounds like A and C. To get around this, draft characters are always printed in double density. In this mode, the firing positions are every 1/120", but any one pin must not fire more often than every 1/60" (i.e., every dot is always followed by a 1/60" gap). For horizontal lines, a pin fires every 1/60". For diagonals, the column of pins fires staggered: each pin fires 1/120" after the pin above (or below), but no one pin fires more than every 1/60". This now gives a matrix of 11 x 8, and you will find that all the capital letters can be drawn quite neatly using the top 7 pins. Why 11 wide and not 12? Simple. Any letter might start with a dot, so column 12, which is 1/120" from the beginning of the next letter, must be kept free in case. You can see now where the risk is of damaging the printer. Standard draft fonts have the dots properly spaced, with at least one rest position between two firing positions; defective software could try to fire the same pin on 1/120" centers, which would damage it.

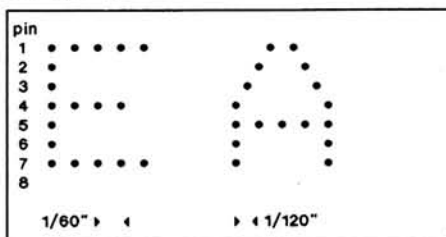


Figure 1-1
8-Pin Capitals on 1/60"
and 1/120" Centers

- So far, we've talked of 8 pins, which is understandable for 8-bit machines. If you try to draw lowercase letters using the 11 x 8 matrix, you will find the size too short for g, p, q and y. This is where pin 9 comes in. One pin on the printhead is always unused for a given letter, usually pin 9. But certain letters can be shifted down one pin, so as to use pins 2 thru 9; this gives you the four letters I just mentioned. There are always 8 pins in use, it's just the spare pin which can be number 1 or number 9, depending on the letter.

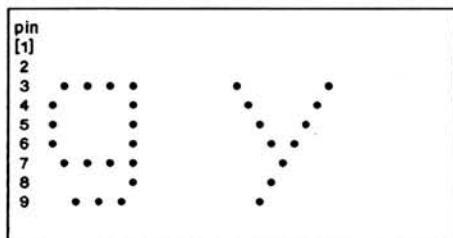


Figure 1-2
The 9th Pin in Lowercase
With a Descender

Letters and Punctuation

Now we're going to get into the differences between FX and GP character sets. The Epson FX printer is an industry standard, designed before the IBM PC came out. Its character set corresponds to the pre-IBM computers: 96 symbols, plus 32 codes, from 0 to 31, reserved for ASCII control characters. 8 bits gives 256 symbols in all; the second slice of 128 characters just repeats the same characters, but in italics. 96 characters is enough for printing in English, but extra letters are required for European languages. These are available in three ways. First, only 13 of the ASCII control characters are actually used. The other 19 can print a short assortment of international characters, upright in the first slice (0 to 31) and italic in the second (128 to 159). Secondly, the 13 control codes between 128 and 159 are not necessary; they already exist from 0 to 31. So you may use all 32 characters from 128 to 159 as printable foreign letters, giving a fair set of italics. This leaves us short of a few upright characters. To get these, several less-used special signs - #, \$, @, [, \ , ^ , ' , { , | , } , and ~ - can be re-assigned to the most useful letters for a particular language, and italicized the same. There are 8 possible re-assignments, one for each major language, plus the basic U.S.A. font.

IBM completely revamped the character assignments when they brought out the PC which was reflected in their new Graphics Printer or GP. From positions 32 to 126, the codes are the same as the Epson FX, which means that for data processing you can get the same results on either printer. Since IBM was going for a world market, they included *all* reasonable accented letters at the same time, grouped together from 128 to 168. The only character the GP doesn't have is an 'o' with four dots around it, in the Swedish set. Of course, using codes from 128 onwards for upright characters means that the GP has no italics, as the corresponding character positions are already in use for something else. So IBM completed the upper codes with extra punctuation marks (169 - 175), semi-graphic characters (176 - 223), a few greek letters (224 - 237) and some mainly math symbols (238 - 254). And at the bottom end (where the FX has the upright accented letters), you

find the four card suits (3 - 6), and the section symbol § (21 on a GP, 93 on an FX).

The IBM semi-graphic characters are nice, but can also be a nuisance. I used them to box in the two figures above. At the standard spacing of 6 lines per inch, they close completely from one line to the next. If you do the sums, 1/6" with pins on 1/72" centers means 12 pins. The printer has to print one pass for the top eight pins, advance the paper 1/9" and print the four remaining pin positions, then advance another 1/18" to the next line. The original GP specifications called for a single line on all boxes, even where the IBM screen character gives a double line like: thus,



Some manufacturers give you single and double lines, others stick to the original specs. Whichever you have, expect the 1/9" partial platen advance to foul up any combination of semi-graphics with bold or underlined characters, line overprints, and mid-line pitch changes.

Epson has since added an extension character set to go some way towards the possibilities of the GP. You may find "Epson Graphics" available as an alternative to the italicized international characters from 128 to 159. The standardized set gives single-line box characters from 128 to 138; gray and black blocks from 139 to 142; and an assortment of symbols for the rest which includes GP ideas like card suits, but also new characters such as a telephone, an airplane, a walking man...

Yes, They Are Compatible

All this doesn't mean that you can't use an FX on a PC, or a GP on non-compatible machines. It just means extra work. Most software doing any word processing now comes with an optional

printer compose table. This normally contains two bytes for each of the 256 characters. If the first byte is non-zero, then it gives a replacement code for the character: thus, for an FX coupled to a PC in France, position 21 will contain '93 0' to show that the section sign § is not at 21, but at 93 (instead of] which isn't on the French set). If the second byte is non-zero, then it gives a composite character: thus, - and u gives ũ. All combinations of circumflexes and diaereses are in the basic GP set, but can only be obtained on an FX by overprinting. Most printer compose tables are easy to patch, as long as you have the two tables of characters for your computer and your printer, and they work well. WordStar 4.0 is more tricky, but it's the best compose table I've seen: there are three bytes per character, giving the two characters to use as usual, and also the FX country character set. The print driver can shift dynamically from one Epson National Character Set to another, to get a very full set of available characters.

If you haven't got a compose table in your software, the attached TSR (Terminate and Stay Resident) program will give you remapping possibilities on any full IBM compatible. It's a simplified version of one I wrote for the Z-158 coupled to an FX and also to a 24-pin printer with GP emulation only. WordStar can handle the mix, but dBASE won't - its two FXs or two CPs for version 3 at least. The table given covers basic mapping for French characters; you can use whatever symbols you wish, but the FX printer must be set to the proper National Character Set before printing.

In the next articles in this series, we'll get down to some brass tacks on just how to obtain these effects - and others!

TSR Printer Driver

After linking this program, convert it to a .COM file with EXE2BIN.

```

title      PRINT_IO          printer communication
                                routine
;
; page      ,120
;
; TSR program to handle printer output
; Includes a compose table for Epson FX codes
;
; -----
;          BIOS RAM reserved locations
; -----
data       SEGMENT           at 40h
.xlist
          DW      4 dup(?)
port_addr DW      4 dup(?)
          DB      104 dup(?)
time_out  DB      4 dup(?)
.list
data      ENDS
;
; -----Printer interrupt routine
;

```

```

code      SEGMENT
ASSUME    CS:code,DS:data,ES:code
ORG       100h

printer_io PROC    far
JMP       install

f01w      LABEL word
pr_sti:   STI       ;driver code starts here
PUSH      DS       ;on input
PUSH      DX       ;DX = printer n°
PUSH      SI       ;AH = operation:
PUSH      CX       ; -0 print AL
PUSH      BX       ; -1 reset printer
PUSH      ES       ; -2 return status
PUSH      DI
PUSH      CS
POP       ES
MOV       BX,seg_data
MOV       DS,BX    ;RAM DOS
CLD
MOV       SI,DX    ;printer n°
MOV       BH,time_out[SI]
SHL       SI,1
MOV       DX,port_addr[SI]
OR        DX,DX
JZ        return ;no printer
OR        AH,AH
JZ        impro   ;0: print
DEC       AH
JZ        init    ;1: init
DEC       AH
JZ        status ;2: status

;
return:   POP       DI
POP       ES
POP       BX
POP       CX
POP       SI
POP       DX
POP       DS
IRET

;
impr:     MOV       DI,offset eptab
MOV       CX,Itab ;compose table
REPNE     SCASB
JNZ       imnorm ;not special
SUB       DI,offset eptab+1
PUSH      AX       ;for later
MOV       AL,ES:epcv1[DI]
CALL      outc     ;remapped char
TEST      AH,1
JNZ       epsab    ;timeout
MOV       AL,0
CMP       AL,ES:epcv2[DI]
JNE       epsbs    ;go to overprint
POP       AX       ;unmapped char
JMP       status

epsbs:    MOV       AL,8 ;backspace
MOV       DX,port_addr[SI]
CALL      outc
TEST      AH,1
JNZ       epsab
MOV       AL,ES:epcv2[DI]
MOV       DX,port_addr[SI]
CALL      outc ;to overprint
TEST      AH,1
JNZ       epsab
POP       AX       ;unmapped char
JMP       status

;
epsab:    POP       CX ;straighten out
PUSH      AX       ;the stack
JMP       stat_2

;
imnorm:   CALL      outc ;straight output
PUSH      AX       ;for later
TEST      AH,1
JNZ       stat_2 ;timeout
POP       AX

;
status:   PUSH      AX
stat_1:   MOV       DX,port_addr[SI]
INC       DX
IN        AL,DX ;get status
MOV       AH,AL
AND       AH,0F8h ;clear bits 0 - 2
stat_2:   POP       DX ;get back char
MOV       AL,DL ;toggle bits 6 et 3,
XOR       AH,48h ;ack et e/o error
JMP       return

;
init:     PUSH      AX ;reset printer
INC       DX
INC       DX
MOV       AL,8 ;00001000,
OUT       DX,AL ;slct/rset/-auto/-strobe
MOV       AX,1000 ;wait printer firmware
att_3:    DEC       AX
JNZ       att_3
MOV       AL,0Ch ;00001100,
OUT       DX,AL ;slct/-rset/-aut/-strobe
JMP       stat_1

;
outc      PROC      near ;print char in AL on
MOV       BL,BH ;restore time_out
PUSH      AX ;port DX
OUT       DX,AL ;latch to data lines
INC       DX ;for status
att_1:    SUB       CX,CX
att_2:    IN        AL,DX ;get status
MOV       AH,AL
TEST      AL,80h ;bit 7=busy
LOOP      att_2 wait not busy before
DEC       BL ;dropping strobe
JNZ       att_1
OR        AH,1 ;stuck busy, timeout
AND       AH,0F9h
;
; and clear bits 1 & 2
JMP SHORT tmout
strobe:   MOV       AL,0Dh ;00001101 strobe low
INC       DX ;latch
OUT       DX,AL
MOV       AL,0Ch ;00001100 strobe high
OUT       DX,AL
POP       AX ;get char back
RET

;
tmout:    POP       BX ;flush AX
RET ;from stack

outc      ENDP

;
; COMPOSE TABLE: replace char in EPTAB
; with EPCV1, overprinted with EPCV2
;
eptab     DB 'àëùéâäïôûäëïöüÿ*§ç'
epcv1     DB 40h,7Dh,7Ch,7Bh,5Eh,5Eh,5Eh,5Eh,
DB 5Eh,7Eh,7Eh,7Eh,7Eh,7Eh,7Eh,5Bh
DB 5Dh,5Ch
epcv2     DB 0,0,0,0,61h,65h,69h,6Fh,75h
Itab      EQU epcv1-eptab

;
end_printio LABEL word
DD 0

;
install:  SUB       AX,AX
MOV       DS,AX ;start of memory
MOV       AX,1
MOV       CS:f01w-2,AX
CLI
MOV       AX,offset printer_io
MOV       DS:(23*4),AX
;
; IP interruption 23 (17h)
MOV       AX,CS

```

Continued on Page 64



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

PC Voice Recognition System with Speech Output

The new Voice Master Key System lets you add voice commands and digitized speech to IBM PCs and compatibles running under MS-DOS. Recognition software is a RAM resident utility (64 Kbytes) that allows up to 256 voiced inputs to replace dozens of keystroke or mouse point-and-clicks with fast and accurate results. The speaker-dependent, isolated word recognizer is suited for CAD, desktop publishing, word processing, or other programs (even games) where voiced input serves to increase user productivity. A Major feature of Voice Master Key is the ease in which voice commands and their associated keyboard macros are assigned. Speech recording feature creates software sound files that can be attached to a voice recognition keyboard macro for verbal response. Also suited for software developers who desire to incorporate speech/sound into programs. Digitized Graphics based editing and compression techniques provide quality speech/sound to

2500 bytes per second. Voice Master Key System consists of half size card, headset, software, and manual. Price \$149.95. Con-

tact: Covox, Inc., 675 Conger Street, Eugene, Oregon 97402. Tel: (503) 342-1271; FAX (503) 342-1283.



"IT'S USED TO HIGH LEVEL LANGUAGE AND CUSSING."



IBM XT in an H/Z-100

Scottie Board W/ZPC \$149.

OPTIONS AVAILABLE

- PC Compatible Serial Port W/Cable \$50
- 2nd Port W/Cable \$45.
- Clock/Calendar \$45.
- No Solder H/Z-100 MOD Kit \$5.

Requires H/Z-100 MS-DOS 768 of RAM, H/Z-100 Modification.

FACTORY NEW...THE REAL THING

- Z-217-1 W/DOCS, HDWE, CABLES, INSTRS
- With 20 Meg ST-225 Drive \$599.
- With 40 Meg MS3650 Drive \$729.

NEW

- Pair of AT Drives - External, Self Contained, Complete
- Uses 1.2 Meg AT Floppy Drives • W/Power, Cables,
- 34 pin to 50 pin adapter, Instructions - \$299.
- 34 pin to 50 pin adapter alone - \$23.95

IBM XT is a registered trademark of IBM Corp. • ZPC is a product of HEATH USER GROUP.

PC/XT/AT and WORKALIKES

- LOGITECH Scanman 4" x 10" Scanner - \$199.
- C7 Serial Mouse - \$ 79.
- HI RES BUS Mouse - \$ 89.
- Complete PC Hand Scanner - \$169.
- GRAVIS Joystick - \$ 39.

HARD DRIVES - All Makes/Models at BEST PRICES

ADD ON CARDS, VIDEO, Controllers, Adapters

Tape Backup 40/60 MEG, 2 MEGS/minute.

Colorado Memory Systems DJ-10 \$299.

VCR Tape Backup "IMAGER" \$199.

BEST PRICES ON:

TURBO XT, 80286, 80386 Computers -

BUYING SERVICE - any item you wish us to purchase COST + 10%
GIVE US AN OPPORTUNITY TO BID ON YOUR NEEDS

Scottie Systems, Inc.

1609 S. Main St.
Milpitas, CA 95035
(408) 262-5021

VISA & MC ACCEPTED, ALL PAYMENTS SUBJECT TO APPROVAL

**ATTENTION
US Government Employees
Special Offer From
Enable Software**



For a limited time, US Government Employees can purchase a copy of Enable for home use at a special price of only \$100.

Enable/FE (2.14) is the same award-winning software program--with word processing, spreadsheet, graphics, database management and telecommunications--that most government employees now use in their office. Enable/FE includes all the powerful features and capabilities you are accustomed to working with day after day, with the exception of Perspective 3-D graphics.

To order your copy, send your check or money order, payable to Enable Software, with your name, federal government ID number, and the address and phone number of your office and home. Mail to Enable Software, Federal Buy, Northway Ten Executive Park, Ballston Lake, NY 12019. Specify 5 1/4" or 3 1/2" disks.

Shipping Costs included for continental US and APO, FPO addresses.

enableFE™
Version 2.14

Reader Service #198

Attention Zenith Z-248™ Users!

**Convert your 286 to a 386 with
our one Board Solution!**

The Aox Z-MASTER 386

**CALL US FOR OUR GSA
SCHEDULE NUMBER!**

800-822-0386

SAI

Strategic Alliance, Inc.
477 Washington Street (617) 762-7485
Norwood, MA 02062

Since 1979 Aox has been a leading designer of accelerator, coprocessor and compatibility cards providing superior technological solutions.

Reader Service #195

REMOVABLE HARD DISK DATA STORAGE



DMA
TECHNOLOGIES

P.O. Box 1236
601 Pine Avenue
Goleta, California 93116

**FOR YOUR ZENITH
Z-150, Z-248, Z-286, Z-386
MICROCOMPUTER SYSTEMS**

- RANDOM ACCESS
- 10MB OR 20MB HARD DISK CARTRIDGES
- MAXIMUM DATA SECURITY
- IMAGE BACK-UP
- 100% MEDIA INTERCHANGEABILITY
- SCSI OR ST506 INTERFACE
- NOVELL COMPATIBLE
- EXTENDED 24 MONTH WARRANTY

For product information see your local Zenith dealer or call 1-805-964-0733.

In California call 1-800-654-8590.

FAX inquiries: 1-805-964-0734.

VISA & MASTERCARD accepted.

Also available on GSA contract GSOOK-88-AGS-5087.

Reader Service #197

BUG

ZAPPING

Bug: From the tests I have made, there appears to be a bug in the PowerHouse software which may or may not have been reported. When this package is used on my H-386, the software will not acknowledge the presence of the X-10 device unless I program the system for SLOW operation instead of FAST operation. That is, the software makes some timing assumptions about the system bus which are lost because of the H-386 speed.

Zapped: My temporary fix is to precede the execution of X10.EXE with:
MODE SPEED SLOW
and to follow the execution of X10 with:
MODE SPEED FAST
Note: Using the Mode command "MODE SPEED SMART" does not fix the problem.
(submitted by C. Gilmore)

Bug: This problem occurs in an H/Z-386 with cache memory board. When selecting the "TEST" function from the MFM-300 monitor, specifically the "Power-Up Test", an immediate display of the error message: "DEFECTIVE TAG RAM CHIP # XXX on CACHE BOARD" appeared.

Zapped: The problem appears to be caused by a bug in older MFM-300 monitor ROMs. The one causing this particular problem was part number 444-549-3. Replace the monitor ROM with the latest version, part number 444-549-8. This also appears to fix the failures that happen with DOSUTIL's ECC test.
(submitted by W. N. Campbell)

Bug: I have a ZWL-200-2(4) SupersPort. What's the latest monitor ROM part number?

Zapped: You need two parts: 444-671-2 and 444-672-2.

Bug: I'm still using Z-DOS on my Z-100 computer. I thought I could read and write standard MS-DOS disks, but can't. In fact, when I tried once writing to an MS-DOS disk, the original information on it was destroyed.

Zapped: Those who use both Z-DOS (or PC-DOS) systems should label their disks appropriately and should be aware of the following potential hazard:

Z-DOS uses an 8 sector per track format. It cannot identify MS-DOS or PC-DOS disks which have been formatted at 9 sectors per track. Attempts to use such a 9 sector per track MS-DOS or PC-DOS disk under Z-DOS can destroy all information on that disk. While merely attempting to display a directory will usually result in an unrecognizable display and will never damage current files, any attempt to write to the disk (i.e., copy files to it) will destroy existing files.

This topic can be quite confusing because of the following 3 items:

1. MS-DOS or PC-DOS versions 2 and higher will read and write to a Z-DOS formatted disk without difficulty yet, the opposite situation, as described above, can destroy a disk.

2. The Z-100 supports an MS-DOS version 2 operating system (OS-61-8) and an MS-DOS version 3 operating system (OS-63-30). Both of these MS-DOS operating systems support the 9 sector per track format that cannot be identified by Z-DOS.

3. Many users mistakenly refer to the Z-100 MS-DOS operating systems (above) as Z-DOS version 2 or Z-DOS version 3.

Because of its severe limitations, it is highly recommended that Z-DOS owners upgrade their software to the latest available version of MS-DOS for the Z-100. This product is model number OS-63-30, and is still available from Heath Company for \$150. However, not many are left, so act quickly!

Bug: I receive the error message "Out of Environment Space" when I try to use the SET command. What's wrong?

Zapped: The environment space has a theoretical limit of 32k bytes. The environment space expands dynamically as it is used, but in actual practice, it is usually limited to about 200 bytes because of memory conflicts with other programs. Any program which becomes resident in memory will prohibit any further significant growth of the environment space allocation. Entering all necessary SET commands at bootup prior to running any program which becomes resident in memory will reserve the needed space in the environment. Since batch files also become temporarily resident in memory, the initial SET commands cannot be entered from a batch file. Environment strings can be removed or modified freely once the maximum needed space has been reserved.

Bug: The keyboard on my Z-180 laptop is more confusing than the flight controls of a starship. Can you clear up some of the fog?

Zapped: The Z-180 PC computer has three keyboard modes: Normal Alphanumeric (Smart), Numeric Keypad and Straight (Dumb).

The Normal Alphanumeric Mode is the mode which exists when the computer is first turned on. This is also known as Smart mode and can be reached at any other time with the FN-1 key combination. This causes all functions of the keyboard to function as labeled.

The Numeric Keypad Mode is activated by the FN NUMLOCK key combination. This sequence causes the keys with orange (Z-181-92/Z-181-93) or blue (Z-183-92/93 and Z-184/SupersPort) labels to generate numeric codes.

Finally, the Straight (Dumb) Mode is the result of pressing the FN-2 key combination. This mode ignores the operation of the SHIFT, FN, NUM-LOCK and cursor control keys. This mode is used mostly for playing games, such as Flight Simulator.

Below are some special notes for the Z-180 PC computer users:

1. Within Lotus version 2.01, Lotus displays information on the screen to tell the user whether or not the NUM-LOCK is active. Lotus will tell the user that NUM-LOCK is active even though the keyboard is in Alphanumeric Mode and not Keypad Mode.

According to the Z-180 PC computer manual: "To prevent accidental operation, the NUM-LOCK key can only be used in combination with the FN key or while in the keypad lock mode . . . Some programs manipulate the keyboard codes by intercepting the up and down codes for each key. These programs can leave the keypad in an unknown state with respect to the keyboard."

Experimenting with Normal Alphanumeric Mode and Keypad Mode gives the following results within Lotus:

When entering or leaving the Numeric Keypad Mode the NUM-LOCK indicator is on.

In Numeric Keypad Mode, the NUM-LOCK key may be toggled on and off with the NUM-LOCK key being pressed. When the NUM-LOCK is off in the Numeric Keypad Mode, the additional five keys -, +, *, =, and / can be used as needed with various software packages.

In Normal Alphanumeric Mode, the NUM-LOCK key may also be toggled on and off with the FN NUM-LOCK key combination.

There are two reasons why the NUM-LOCK indicator appears on the screen even though the NUM-LOCK is not active.

1. The Z-180 series computers toggle the NUM-LOCK bit of the Keyboard Status byte while in Normal Alphanumeric Mode.

2. The Z-180 series computer sets the NUM-LOCK indicator on whenever leaving Numeric Keypad Mode and entering Normal Alphanumeric Mode.

While in Numeric Keypad Mode, the NUM-LOCK information displayed on the screen by Lotus should be correct. While in Normal Alphanumeric Mode, the NUM-LOCK information may be incorrect.

2. Within Framework II version 1.1, users have reported difficulty using the orange/blue "+" and "-" keys.

On the Z-181 and Z-183, in order to use those keys, press the FN key and while holding it down, press the NUM-LOCK key once, release the FN key and press NUM-LOCK once more. Next, press the desired "+" or "-" and press the NUM-LOCK to return to the Numeric Keypad Mode. Or press the FN key and then the spacebar, holding the FN key choose the desired option, "+" or "-". To return the keyboard to Normal Alphanumeric Mode, press the FN key and while holding it down, press the NUM-LOCK key once.

On the Z-184/SupersPort, in order to use those keys, press and hold down the FN key and the NUM-LOCK key once more. Then press the desired "+" or "-" keys. To return the keyboard to Normal Alphanumeric Mode, press the FN key and while holding it down, press the NUM-LOCK key once. To return to alphabetic mode, press and hold down the FN key and the NUM-LOCK key.

Now that the "fog" has lifted, I'm sure it won't be mist!

Bug: Is is OK to pass my laptop computer through the X-Ray machine at the airport?
Zapped: Yes, you can safely pass your laptop computer through X-Ray devices used at airports.

Bug: In some instances, the parallel port on the Z-159 memory board (p/n 181-7578) will fail.

Zapped: A foil on the board shorts to the metal shield of the parallel port connector at P401. This problem has been corrected on the p/n 181-7578-IF and later models. To correct the problem, cut both ends of the foil between U419-11 and U453-9, and connect a small jumper wire between the same two pins.

Bug: In the Z-248/2200 series computer, installing an XT type winchester controller board will cause the error message: "+++ DISK ERROR: Drive not ready +++". The system will also fail to work with newer type ESDI disk controllers and drives.

Zapped: Replace the monitor ROMs. The latest version is currently Ver. 2.5 (p/n 444-423-10 and 444-424-10). The new firmware includes ESDI drive support. If a DTC ESDI Winchester disk controller is installed, SETUP will only toggle between Drive type 100 and -Not Present-. When type 100 is selected, the drive parameters are obtained from the controller and SETUP will display this information for the drive.

Bug: The ZWL-200-2/4 Laptop computer operates improperly when running OS/2 or XENIX.

Zapped: Check if the SLOW mode of operation is being used. The ZWL-200 computer operates properly in the SLOW mode only when using the MS-DOS operating system. When using a protected mode operating system such as OS/2 or XENIX, place the computer in the FAST mode.

Bug: The older version (2.1) monitor ROM caused several problems. 1) The wrong oscillator clock was selected in the CRT mode, causing the external CRT to be out of sync. 2) When switching palettes with FN-F8, it worked until palette zero was reached, then nothing happened. 3) Cursor visibility was poor.

Zapped: Changed the monitor ROM to version 2.7A (p/n 444-671-2 and 444- 672-2). The above problems should be remedied. *

Continued from Page 60

```
MOV     DS:(23*4+2).AX
        CS interruption 23 (17h)

STI
MOV     DX,offset end_printio
INC     DX
INC     DX      ;for luck
INT     27h     ;terminate and stay
printer_io ENDP      ; resident
code     ENDS
        END     printer_io
```



Want New And Interesting Software?
Check Out HUG Software



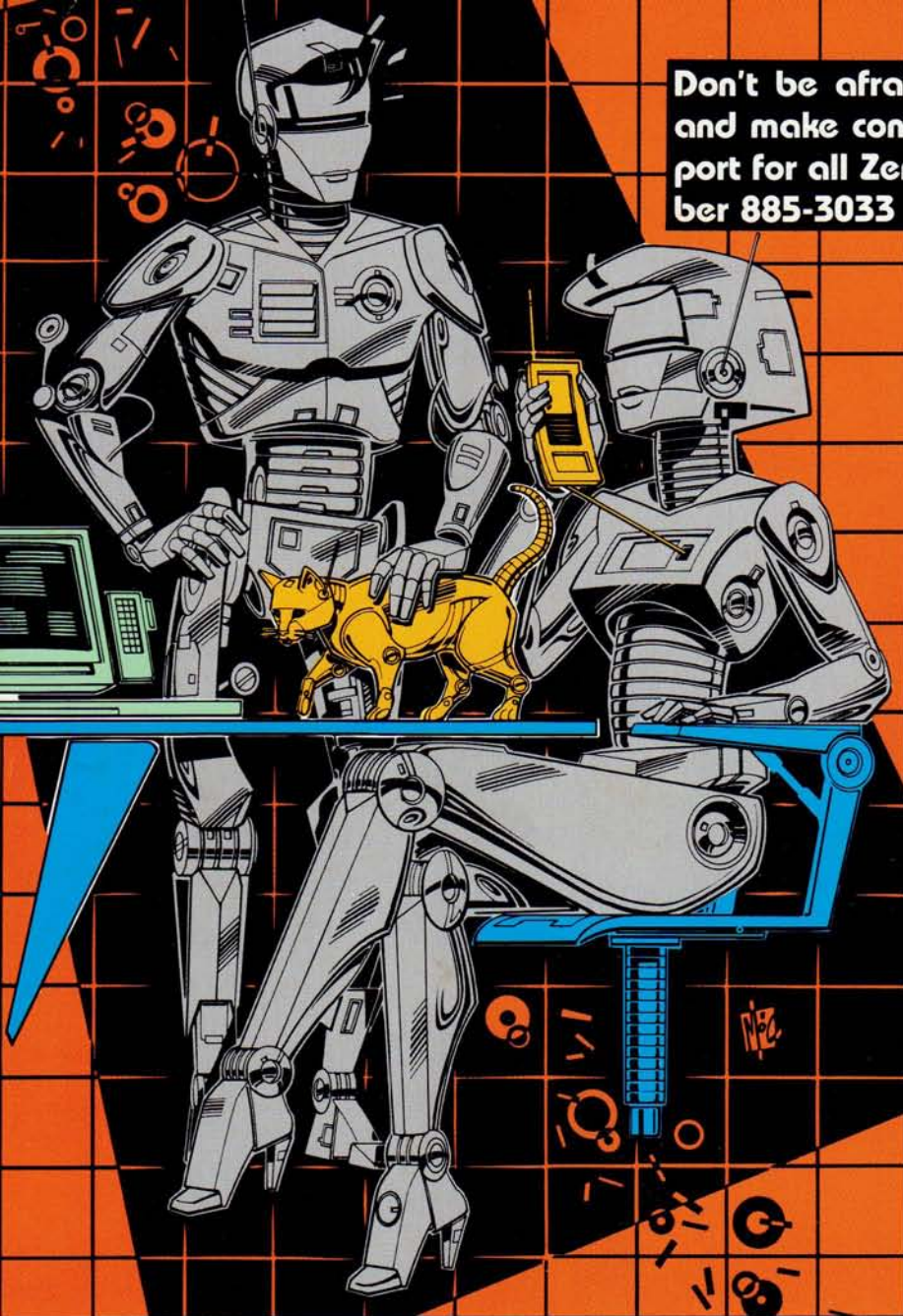
HADES II

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.

Don't be afraid to communicate! Get HUGMCP and make contact the easy way. Now with support for all Zenith 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 (A-0), (F1) To List, Anything Else To Abort --> _

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

HUGMCP Configuration Help #1

1. This Function Allows The Baud Rate To Be Changed. Depending Upon Which Mode You Are Using. Normally, It Would Be Set To Either 9600, 19200, Or 38400 Baud. Select Connection To A Host. Will Allow Higher Baud Rates.
2. This Function Allows You To Change The Word Parity. Normally, you Should Choose No Parity. This Is Acceptable By Most Software Systems. And It Is Also Necessary For XMODEM Protocol To Work Properly.
3. This Function Allows The Changing Of The Word Length. Normally The Length Would Be Set To 8 Data Bits. This Value Is Acceptable By Most Software Systems, And Is Necessary For XMODEM Protocol To Work Properly.
4. This Selection Allows You To Enter Messages Which Can Be Automatically Sent With The (F1) Key. Up To 14, 70-Character Messages Can Be Listed. Selection 14 Is Special. It Should Contain Your Computer's IP Number And Network Selection (0) Is Also Special. This Selection Can Help Materially Be Sent When This Program Is First Executed By Selecting The Proper Option During Setup.

Type (F6) (S) For More Help, Anything Else To Configure.

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

HUGMCP Configuration Menu:

- A --> Modify Baud Rate
- B --> Modify Parity Type
- C --> Modify Word Length
- D --> Modify Or Add Auto-Messages
- E --> Miscellaneous Functions
- F --> Change Screen Color Assignments
- G --> Display Current Configuration
- H --> Make Changes Permanent

Select A-G, (F1) For Help, Anything Else To Quit --> _

Baud Rate: 19200
Parity: NONE
Word Length: 8
Baudline: FULL
Response To Keyboard Disable: NO
Storage Buffer Data Parity Bit: SET TO ZERO
Send Modem Initialization Text: NO
Baud Character: W00000
Modem Port Set To: COM1

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



P.O. Box 217

Benton Harbor, MI 49022-0217

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

POSTMASTER: If undeliverable,
please do not return.

\$2.50
P/N 885-2113