The Official ZENITH/Heath Computer Users Magazine

REMark®

July 1989

A Look at PC Tools Deluxe Version 5

Page 7

New Laptop Utilities Disk

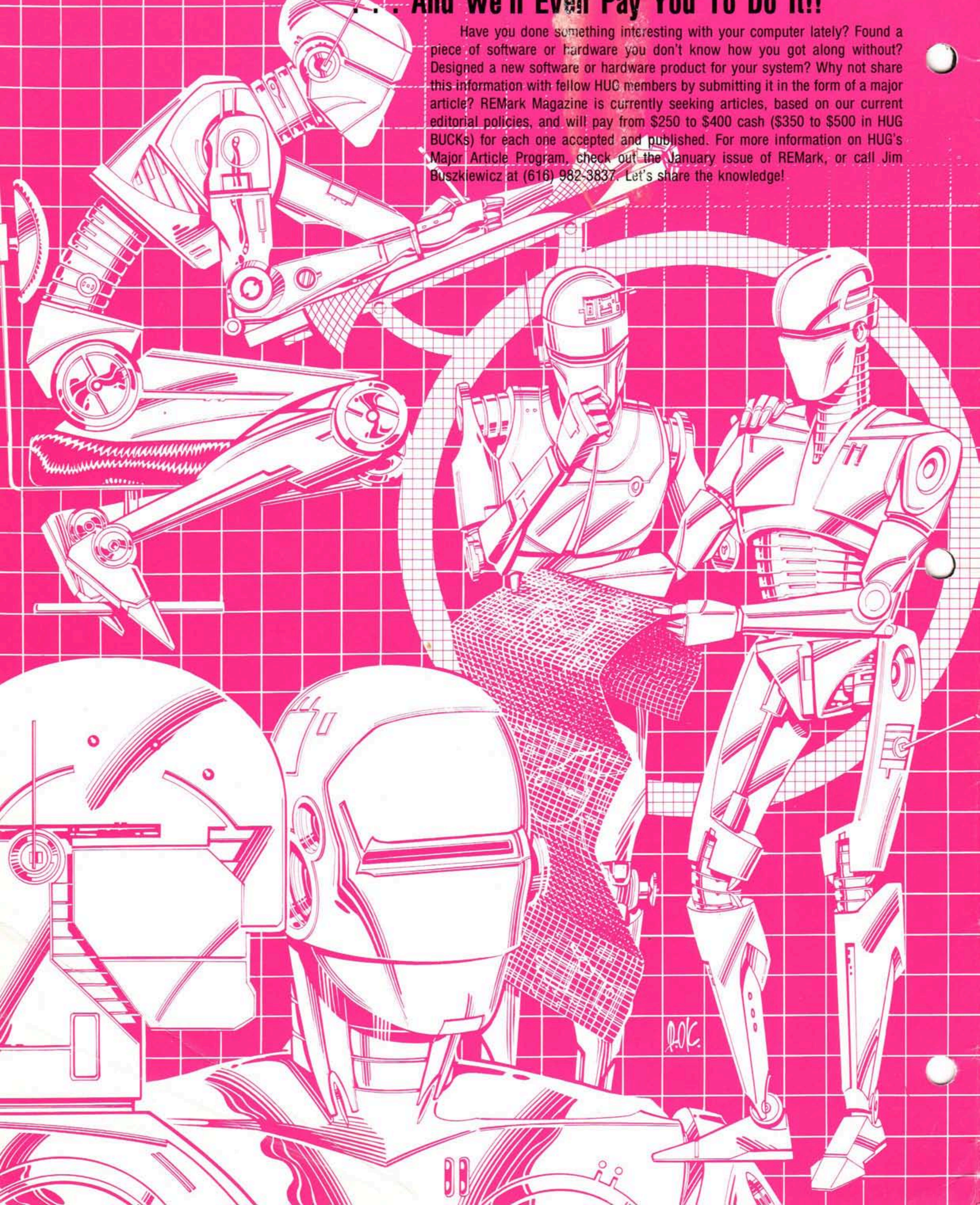
Page 4

Inexpensive Software for the Z-100

Page 51

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!



The Official ZENITH/Heath Computer Users Magazine

REMark®

Volume 10, Issue 7 • July 1989

General

Resources

Reader Service No.		Page No.
101	ACI, Inc.	41
196	Covox, Inc.	29
197	DMA Technologies	56
198	Enable Software	29
104	FBE Research	26
107	Paul F. Herman	11
137	Jay Gold Software	35
136	Lindley Systems	54
114	Micronics Technology	50
117	Payload Computer	12
***	Quality Assurance Ad	64
121	Scottie Systems	45
153	Surplus Trading Corporation	64
194	Weltec	44

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

H/Z-100 and PC Compatible

H/Z-100

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.

HUG

H8 - H/Z-89/90

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
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
AUTOFILE (Z80 ONLY)	885-1110	HDOS	DBMS	30.00
BHBASIC SUPPORT PACKAGE	885-1119-[37]	HDOS	UTILITY	20.00
CASTLE	885-8032-[37]	HDOS	ENTERTAINMENT	20.00
CHEAPCALC	885-1131-[37]	HDOS	SPREADSHEET	20.00
CHECKOFF	885-8010	HDOS	CHECKBOOK SOFTWARE	25.00
DEVICE DRIVERS	885-1105	HDOS	UTILITY	20.00
DISK UTILITIES	885-1213-[37]	CPM	UTILITY	20.00
DUNGEONS & DRAGONS	885-1093-[37]	HDOS	GAME	20.00
FLOATING POINT PACKAGE	885-1063	HDOS	UTILITY	18.00
GALACTIC WARRIORS	885-8009-[37]	HDOS	GAME	20.00
GALACTIC WARRIORS	885-8009-[37]	CPM	GAME	20.00
GAMES 1	885-1029-[37]	HDOS	GAMES	18.00
HARD SECTOR SUPPORT PACKAGE	885-1121	HDOS	UTILITY	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 TRANSCEIVER	885-8016	HDOS	AMATEUR RADIO	20.00
MORSE CODE TRANSCEIVER	885-8031-[37]	CPM	AMATEUR RADIO	20.00
PAGE EDITOR	885-1079-[37]	HDOS	UTILITY	25.00
PROGRAMS FOR PRINTERS	885-1082	HDOS	UTILITY	20.00
REMARK VOL 1 ISSUES 1-13	885-4001	N/A	1978 TO DECEMBER 1980	20.00
RUNOFF	885-1025	HDOS	TEXT PROCESSOR	35.00
SCICALC	885-8027	HDOS	UTILITY	20.00
SMALL BUSINESS PACKAGE	885-1071-[37]	HDOS	BUSINESS	75.00
SMALL-C COMPILER	885-1134	HDOS	LANGUAGE	30.00
SOFT SECTOR SUPPORT PACKAGE	885-1127-[37]	HDOS	UTILITY	20.00
STUDENT'S STATISTICS PACKAGE	885-8021	HDOS	EDUCATION	20.00
SUBMIT (Z80 ONLY)	885-8006	HDOS	UTILITY	20.00
TERM & HTOC	885-1207-[37]	CPM	COMMUNICATION & UTILITY	20.00
TINY BASIC COMPILER	885-1132-[37]	HDOS	LANGUAGE	25.00
TINY PASCAL	885-1086-[37]	HDOS	LANGUAGE	20.00
UDUMP	885-8004	HDOS	UTILITY	35.00
UTILITIES	885-1212-[37]	CPM	UTILITY	20.00
UTILITIES BY PS	885-1126	HDOS	UTILITY	20.00
VARIETY PACKAGE	885-1135-[37]	HDOS	UTILITY & GAMES	20.00
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)

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
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	MSODS	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 Veternary 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.

HUG NEW PRODUCTS

10 – Very Good
9 – Good
8 – Average

TABLE C
Product Rating

Rating values 8–10 are based on the ease of use, the programming technique used, and the efficiency of the product.

- **7 – Hardware limitations (memory, disk storage, etc.)**
- **6 – Requires special programming technique**
- **5 – Requires additional or special hardware**
- **4 – Requires a printer**
- **3 – Uses the Special Function Keys (f1,f2,f3,etc.)**
- **2 – Program runs in *Real Time****
- **1 – Single–keystroke input**
- **0 – Uses the H19 (H/Z–89) escape codes (graphics, reverse video)**

Real Time — A program that does not require interactivity with the user. This term usually refers to games that continue to execute with or without the input of the player (e.g., 885–1103 or 885–1211[–37] SEA BATTLE.

ORDERING INFORMATION

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

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

NOTES

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

All special update offers announced in REMark (i.e., ZPC II update) must be paid by check or money order, payable to the Heath Users' Group. **NO CREDIT CARDS ACCEPTED.**

P/N 885-6014
HUG Laptop Utilities . $20.00

This disk contains a collection of utilities designed to make life with your laptop more enjoyable and productive. Included on this disk are utilities to make your cursor more visible, your graphics better looking, and your keyboard friendlier.

Requirements: These utilities will run on any Zenith or Heath laptop computer and should work on most other brands. Some of the programs utilize features that are unique to Zenith and Heath models. MS-DOS version 2 or above is required.

Program Author: Patrick Swayne, HUG Software Engineer

Program Content: Here is a description of the programs on this disk:

CURSOR.COM — This program provides different ways to make your cursor more visible. You can make the cursor bigger, make it blink more slowly, or make it a non-blinking block cursor. You can also disable this utility if it should conflict with a particular program, and then re-enable it later. **Note:** Non-Zenith laptop computers may not support slow blinking (a function of the Zenith hardware), or the non-blinking cursor (maintained by this program).

REVSCRN.COM — Your laptop screen was designed to make text displays look better, but graphic displays sometime come out looking like photographic negatives. REVSCRN allows you to instantly reverse the shade tones on your screen (just press Ctrl-Shift R), so that your graphics will look like they should. REVSCRN works while the screen is in text or (CGA compatible) graphics modes.

LAPSTAT.COM — Certain Zenith and Heath model laptop computers support an extension to the ROM BIOS called ZBIOS. LAPSTAT uses ZBIOS to provide certain information about your computer, including the internal modem status, the processor speed, the LCD palette, the Zenith code name for your computer (such as DRAGON or BADGER), and the ROM version number. The computer models that support ZBIOS (as of this writing) are the SuperSport 286 and the TurboSport 386.

CAPCON.COM — This program allows you to reverse the functions of the Caps Lock and left Ctrl keys on computers that do not provide a reversal switch. The keys can be swapped back to their default positions when required.

KEYS.COM — With this utility, you can set up the default state of Caps Lock, Num Lock, and Scroll Lock from the command line or in a batch file. You can make your computer boot up with the keys the way you want them.

D.COM — This is the HUG alphabetizing, columnizing directory program, always a handy thing to have on any computer.

DT.COM — This program prints a graphics tree display of your directories. Great for hard disk users.

F.COM — With this utility you can locate a file on a large disk without having to search each directory individually.

HFM.COM — This is an improved version of the HUG File Manager, a utility that makes it easy to keep up with the files on a large disk. HFM can copy or delete selected groups of files or individual files, rename files, create or remove directories, label disks, and more.

SEE.COM — This utility is a fancy replacement for the TYPE command for examining text files. You can move forward or backward in a file, search for text, and print selected text.

DTEST.COM — This is a non-destructive disk testing utility that can be used anytime you want to check the condition of a disk (hard or floppy). If you travel around with a hard disk laptop, you can check your hard disk after each trip. DTEST can also mark bad sectors that it

Continued on Page 20

BUGGIN' HUG

DIR in Two Columns

Dear HUG:

I enjoyed Mr. Lisanti's article on Multi-Edit and am only sorry that the address of American Cybernetics was not in the article. (I have now got the address.) I should like to see more articles on advanced programming.

I picked up the following tip from an article in the UK IBM Users' magazine. It concerns the DOS DIR command. As you know, DIR only uses half of the screen, i.e., columns. It is very easy to have DIR use the full 80 columns and thus display two columns of files on the screen instead of only one. A simple change to COMMAND.COM is all that is needed. Using Norton, ZDUMP or Debug, etc., search for the string A8 01 B0 01 74 02 B0 05 and change it into the following: A8 01 B0 **02** 74 02 B0 05. This one byte change will give you two columns of files on DIR after the next reboot. This works on the Zenith 3.20 version of command.com. Perhaps you already know about this, but it has not appeared in any REMark that I have seen.

Yours faithfully,
J. B. Gibbons
Strickland Manor Farm
Yoxford, Saxmundham
Suffolk IP17 3JD
ENGLAND

***Ed:** This mod works fine with MS-DOS Version 3.3 Plus. Also, HADES is ideal for making this change, and remember to reboot for this to become effective.*

SWFHUG Disbanded

Dear HUG:

Please note in one of the next issues of REMark that the SWFHUG group I started in 1981 has been disbanded and most of the members joined up with the local PC users group. I am still available to help anyone that has an H/Z-100, H/Z-89 at the same phone number as in the previous ad for the club, but at the address that is shown here, not the P.O. Box in Tice, FL. In the time from the spring of '81 to the middle of last year, I published a monthly newsletter with no help from any of the members, except when Paul Herman was President of our group before he moved up state. It seems that the membership expected the newsletter every month, but no one wanted to help. I got the info together, wrote it up and got it printed. We had several hams in the group so I had a column for PACKET and RTTY. The last president we had talked the ones in the group that had the compatibles into going to the PC group so that took about all the active members from our group.

Robert W. Sloat
P.O. Box 1319
Fort Myers, FL 33902-1319

Your First Hard Disk Failure

Dear Mr. Rogers:

I enjoyed your article entitled, "Your First Hard Disk Failure" in the May 1989 "REMark" magazine. I've been there before!

Near the end of page 9 you mentioned that you were looking for a utility which would copy to a floppy any file that was created or changed that day, but not in archive format. What I am recommending doesn't do this automatically, but I find works well for me. I use the memory resident utility WindowDOS (version 2.0), which allows you to sort the directory in any order you choose. I have it set up by default to sort the directory alphabetically by extension, then (for all with the same extension) alphabetically by file name.

The program allows you to change the sort order at any time and also allows you to mark the file names (with a "+" plus sign) which you would like mass copy to another drive/directory. My daily backups are easily handled by resorting the directory by creation date in descending order (which places all of today's files at the top of the list), hitting plus (or holding down the "+" key until all of today's files are marked with a "+" next to them), hitting "C" for copy, then typing the drive/directory I want the files copied to. Another shortcut allows me to, within WindowDOS, "log to" the destination drive/directory before the copy operation, and then I only need to specify the drive the files should be backed up to and they go to the "logged to" directory.

One advantage of using WindowDOS is being able to back up files created within any recent time period I choose (for instance, 2 days, the current week, this month, etc. — whatever file dates I identify with the plus key). In addition, if I keep my data files in a separate subdirectory from my program files, I can mark ALL data files in it for back up by hitting the "insert" key (this puts "+" marks procedure can also be used to mass erase old files. An additional advantage is that should your backup floppy disk fill up, the WindowDOS program will prompt you to insert a new backup disk, then continue to copy the remaining programs.

The only disadvantages of using WindowDOS that I can think of are the small memory penalty of 52.3K and the fact that you have to back up one directory at a time. These haven't been problems for me because (1) I cannot imagine not having WindowDOS readily available (you can operate it on a NON-memory resident program and even uninstall it from memory, once loaded) and (2) I like to keep my backup data files in subdirectories on floppy disks which have the same name as the subdirectories on the hard disk from which they are copied. This makes it very easy to compare files, as well as reload subdirectories, if need be.

For more on WindowDOS see page 28 of the May 1989 REMark and the September 1987 REMark (page 34 and page 83) and/or write WindowDOS Associates, Box 300488-B, Arlington, TX 76010. Phone (817) 467-4103. It cost $49.95 when I bought it.

This sounds like an advertisement, but I honestly don't know what I'd do without WindowDOS, it makes life so much easier when you have a hard disk. Good luck, and thanks for your useful article in the May 1989 REMark.

Sincerely,
Bob Schornstheimer
210 Aikane Street
Kailua, HI 96734

Your First Hard Disk Failure

Dear HUG:

Regarding Bill Rogers' aritcle, "Your First Hard Disk Failure," I'd recommend FASTBACK as a good, general purpose backup utility.

I've been using FASTBACK for about two years on my PS/2 at work and it has served me well. (I can't use it at home since IBM chose not to maintain compatibility with the H8. An oversight, I'm sure.)

FASTBACK comes with two major utilities; FASTBACK for backup and FRESTORE to restore files. When you start FASTBACK, it will ask which disk to backup (default: C), which directory (default: root), whether to backup subdirectories (default: yes), and whether to skip files that haven't changed since the last backup (default: no). This 'skip' feature keys off the ARCHIVE bit in the directory.

FASTBACK uses it's own floppy disk format and will format raw disks as it goes. The first time you use a given floppy, the backup takes a while due to the formatting. On subsequent backups it really moves.

FRESTORE allows you to restore individual files or all files. To restore an individual file, you point to the directory/file on the screen and type 'R'. To restore all files, you type Ctrl-R.

When you use the 'all files' option, you can choose whether to replace existing files, skip over existing files, or replace after prompting the user. If the directory is missing, FRESTORE will restore the directory before restoring the file.

FRESTORE also has a 'verify' option that will compare each file on the backup with its counterpart on the hard disk, so you can check the accuracy of the backup.

FASTBACK leaves a file catalog on the hard disk, used by FRESTORE to allow you to quickly spin down to the file you want to restore. I had problems running FASTBACK until I discovered that you have to delete this catalog before you run the utility. Otherwise, FASTBACK tends to raise a "wrong disk" error and become quite uncooperative. Note that if the catalog is lost (as it would be if you lose the whole hard disk), you can use the catalog recorded on the backup floppy disks. Directory searches are slower on the floppy-based directory, but the functionality is the same.

I've never tried the 'skip files that haven't changed' option. I don't really think I'd like to keep track of files that way. You wouldn't be able to reuse a backup disk until you took another complete backup. If you need to restore a file, it would be hard to tell which disk contained the file.

I've had considerable success with the following approach to disk organization. I create two directories off the root: PRGS and WORK. PROGS contains miscellaneous programs. I also create a subdirectory off PROGS for each major software package. WORK (and various project-related subdirectories) contains all work files, those that change daily. It looks like this:

```
\
¦
¦
+---PROGS
¦   +---DOS
¦   +---123
¦   +---HARVARD
¦   (etc.)
¦
+---WORK
    +---MKTPLAN
    +---LETTERS
    ¦   +---CUSTOMERS
    (etc.)
```

I stick all my .BAT files (except AUTOEXEC.BAT) in PROGS. The root contains only AUTOEXEC.BAT, CONFIG .SYS, COMMAND.COM, and a couple of required device drivers. My PATH statement is C:×PROGS;C:×PROGS×DOS. Everything else is run using a .BAT file.

I have not run into a single commercial program that can't support this directory scheme.

I will periodically take a full backup of the hard disk. I'll do this after installing a new or upgraded package, or when I change the configuration of something. Then, more often, I'll take backups of ×WORK and all its subdirectories. I keep three or four floppy disks for this purpose and rotate them. (I note in my desk calendar which disk was used last.) I'll backup ×WORK weekly or, say, at the end of the day after a long editing session.

I have had to do a full restore only once, and it worked like a charm. I would note that I used this same disk organization scheme with the BACKUP and RESTORE utilities provided by DOS and it worked just as well, except for the clumsiness of the utilities themselves. The whole thing works better with FASTBACK.

FASTBACK is available from:

Fifth Generation Systems, Inc.
11200 Industriplex Boulevard
Baton Rouge, LA 70809

You should also be able to find it in any computer store.

Sincerely,
David A. Shaw
11059 Overrun Drive
Manassas, VA 22111

Original CRT Saver for Z-100 (NOT PC)

Dear HUG:

I hate to bring the CRTSAVER visits up once again, but as a result of those articles, I am beginning to develop an inferiority complex.

When I decided it was time to get down to serious assembler language coding, I purchased the utility package that contains all the goodies except the Microsoft Assembler Reference manual. I felt pretty confused, so I purchased some manuals with sample programs and reviewed my back issues of REMark.

After assembling a couple of simple routines and getting them to work, I decided to do the now famous CRTSAVER utility program. I started with the October 88 version by W. T. Vomocil. Except for a few typographical errors (I figured out there were some delimiters missing, specifically the framing characters for literals, probably quotation marks?) I was able to get a clean assembly. However, upon linking it, I was unable to get it to work, so I decided I would use the March 88 version by R. G. Brasfield. After meticulously checking every single byte, I assembled and linked it. You guessed it, the results were even worse than before, this time I get several happy faces and other gambling paraphernalia on the screen followed by the DOS prompt.

I have run the assembly and link on two PCs, a Zenith 138 and an IBM XT, with similar results. I am beginning to wonder if this is just a ruse to get HUGGIES to pay attention for a pop quiz later! Or did I miss something important, like it only runs on Z-100 (NOT PC).

I have not given up yet, but I am pretty close to it. Maybe I'll just stick to something simple, like IBM/370 Assembler or mainframe ANSI COBOL.

Resp'y,
Manuel Campos
2728 Greenhill Drive
Mesquite, TX 75150

The original Screen-Saver program that generated all the "CRT Saver Revisited" articles, originally was written by Frank Clark, and appeared in the July 1984 issue of REMark and yes, it is for the Z-100 (NOT PC)! Sorry, Jim.

Dr. Cole's Bug

Dear HUG:

Regret that I am able to confirm Dr. Harry Cole's MS-DOS 3.3 Plus bug ("Buggin' HUG", REMark, May 1989, page 6) using my H-151. My operating system, however, is Zenith's OS-51-1.

Not only is the F3 key (repeat entire last command) affected, I also found the F1 key (repeat last command character by character) and the F2 key (press F2 and a character — last command up to, but not including that character, is repeated) to be equally affected.

I am certain that I have seen Dr. Cole's problem in the past. Until publication of his finding, however, I blamed the operator for too light a keystroke!

Since a large number of us use these keys with their built-in macros, any fix by Zenith or anyone else, will be appreciated.

Cordially,
John R. Miller
401 Tiffany Drive
Anderson, SC 29625-1815

Expanded Memory and Partitions

Dear HUG:

I originally purchased a Z-159 with Dual Disk Drives. I later decided to add a Hard Disk, so I replaced drive B with a Seagate ST251 drive. This is a 40 Megabit unit. While I was able to format it for one 32 Megabit partition I was unable to set up a second partition. This gave me Drives A and B for the floppy, C for the hard disk and D for the expanded memory.

The first article in "Bug Zapping" on page 63 of the May '89 issue of "REMark" addressed this problem and with its help I was encouraged to try again. Everything went well and I now have the balance of the drive set up as drive D. I also changed

Continued on Page 35

Getting Started With . . .
PC Tools Deluxe, Version 5

EVEN THE KITCHEN SINK!

Richard J. O'Connor
848 Fenske Drive NE
Olympia, WA 98506

Introduction

Admit it. You've been tempted.

Originally, you were going to limit your software purchases to specific applications, right? Word processing, communications, programming languages, and (for that critical hand-eye coordination) games; that's all you needed.

And then "productivity enhancement" software was born, a category you'd never heard of. DOS shells, tree displays, pop-up calculators, tiny editors; all a keystroke or two away. As these extra applications became more integrated and just plain handier to use, maybe you began to feel your resolve slipping away . . .

I've been working with Heath/Zenith microcomputers on the job since I helped a fellow worker put together our agency's first Heathkit H-100 back in 1984. Serious, JUSTIFIABLE applications, that's all we bought for that machine. Of course, there wasn't much else on the market, either! But when Cathy and I bought our first home computer, a twin floppy Zenith Z-159, in the spring of 1987, this "new stuff" was popping up and out all over the place. Still, we stuck to our original game plan; PC WatchWord, QuickBasic, Turbo Pascal, First Publisher, and a couple of Sierra games were all the applications we needed. If we wanted something else, we wrote it. Sound familiar??

We resisted the urge to explore this new software category until last fall, when we installed a 30 megabyte hard disk and I began to think about getting "some hard disk utilities", like those offered by Peter Norton and Paul Mace. But the cash register in my mind rebelled at the mounting costs . . . did we really need ALL the power those packages had to offer? Maybe all we needed was a good unfragment program, some undelete tools, a tree-oriented shell for easier file/directory maintenance, and one of those speedy disk caching programs. For now, anyway!

Last fall we received a flyer announcing Version 5 of PC Tools Deluxe, and the combination of features offered made me wonder if they'd been reading my mind! The list went far beyond our immediate needs, but the real clincher was the price. Suggested retail price was $79, but previous Central Point customers could order a copy for the upgrade price of $15! There went my last excuse; we ordered it.

After some delays in final release date, we received our copy in late January. We've been exploring the package ever since, and in the next few installments, I'll share our findings with you. I'll start by listing the eight programs that make up PC Tools Deluxe, Version 5 and walking through installation with you. Our guided tour continues with an introduction to some of the many uses of PC Shell. I'll close with a look at the PC Format program, and explain why it's a better choice than the format program provided with MS-DOS. Please note that the current version of PC Tools Deluxe is Version 5.1, but that this article was written about Version 5.0. Why are so many users, particularly newer users, attracted to MS-DOS "shell" programs? I think there are two main reasons that such programs are very helpful to MS-DOS newcomers. The first reason is that MS-DOS is not an intuitive system, and the strict format of MS-DOS commands is difficult to remember. Shell programs hide details such as command format from you by prompting you for all of the specific information needed when a command is selected. The second reason is that MS-DOS is an unforgiving spelling teacher, and will beat you over the head with "Bad Command or File Name" for a simple misspelling of a command. A shell program allows you to select the command you want from a correctly spelled list, confident that it will run without complaint. MS-DOS shells are handy tools for advanced users as well (some of us can't spell either!), mostly due to the way they organize the available commands into logical groupings.

Hopefully, this article will help you to decide whether your software collection is ready for an inexpensive and very powerful addition. Even if you think "productivity enhancement" software is more of a sales pitch than a description, read on; you may be quite surprised!

Just What Is PC Tools Deluxe?

PC Tools Deluxe (PCTD) is a package containing eight separate programs that provide a considerable array of tools for the DOS environment. The eight programs are PC Shell (a tree-oriented shell program with complete file and directory maintenance), PC Tools Desktop (desktop manager including editor, spelling checker, data base, calculators, communications program, etc.), PC Backup (a fast diskette or hard drive backup utility with optional data compression), Compress (file unfragmenting, subdirectory sort and move, hard error marking), PC-Cache (caching program to speed up disk reads), PC Format (special formatting tool that enhances data and file recovery), Mirror/Rebuild (protection against hard disk reformatting, full recovery of deleted files), and PC Secure (file compression, plus DES encryption and decryption of data files in the domestic version only).

The consistent and flexible user interface is possibly the strongest point of this package. PCTD provides full mouse support for its pulldown menus, so those of you familiar with mice and Microsoft Windows-style pulldown menus can rely on intuition to navigate the familiar scenery of PCTD screens. Users without a mouse can select options using keyboard sequences, along with certain function key shortcuts which cut down on the number of keystrokes required.

But, the cautious among you are asking, how do I back out when I've wandered down a path of menus and I'm not sure how to get back home? PCTD provides nice, logical escape routes once you're finished with an application. Mouse users can "put a window away" by clicking in the small box drawn at the upper left-hand corner of each on-screen window. Also, highlighted Exit or Cancel boxes appear when a choice is needed, and when all else confuses, the venerable Escape key will send you back one level. No matter what you're up to, you can gracefully back down before you destroy anything inadvertently. PCTD seems to have been programmed with the fatally curious (me) in mind!

The PC Shell and PC Tools Desktop programs give the user complete control over the appearance and placement of the windows they display. Most windows can be moved, resized, reshaped, and recolored to suit your preferences. This is very handy, once you realize that up to 15 windows can be open on your screen at once in the Desktop program! My favorite part of the windows implementation is the "shadow" each window casts on lower windows. This graphics effect makes it quite clear which window is "active" (in the foreground) if you get carried away!

The documentation for PCTD Version 5 consists of a 500-page manual, a 24-page printed Addendum, and a READ-ME.TXT file with last-minute corrections. The manual steps through each application in a clear fashion, and repeats key concepts (such as selecting options) whenever they come up in each section. Use of this approach, rather than the See Page 23 (Again!) approach cuts down on the page-thrashing other manuals seem to delight in putting you through. Sample screens are easy to read, and there are enough examples to clarify the use of any of the tools. In short, they did a fine job on the documentation, which gave me high hopes for the tools themselves!

In the most recent release (Version 5.1), the documentation has been repackaged into three manuals, which total over 600 pages. One volume covers the PC Shell and data recovery programs, one covers the PC Tools Desktop program, and one covers the hard disk utilities.

Installing PC Tools Deluxe On A Hard Disk

You don't need a hard disk to run PCTD, but many of the utilities appeal to hard disk users; I assume many purchasers will fall in this category. Central Point Software lists the requirements for Version 5.1 as follows: an IBM PC, XT, AT, PS/2 or compatible, 512K of memory and DOS 3.0 or greater. (Note that Version 5 required DOS 2.0 or greater; I tested this version using MS-DOS 3.1.) To run PC Shell and PC Desktop in resident mode, 640K of memory is recommended. PCTD will run in monochrome, but it's a lot more fun working with a good EGA or VGA monitor, like my Zenith 1380C. A mouse makes working with PCTD much easier, though you'll need Microsoft mouse driver version 6.14 or greater or Logitech mouse driver 3.4 or greater in order for all applications to work correctly. Our Z-159 has 1 megabyte of memory (340K set aside for EMS, which can be used quite effectively by PCTD) and runs MS-DOS version 3.10 with a Microsoft mouse and driver version 6.14. If you would like to use your mouse, but your driver is a little out of date, simply call Microsoft Customer Support at (206) 454-2030 and ask for Hardware support. They will send you a copy of the most recent mouse driver, and you're all set!

PCTD is provided both on standard 360K diskettes (4 of them!) and on 2 3.5 inch 720K diskettes. A program called PCSETUP is provided to automate installation of PCTD on your hard disk. PCSETUP does several things you should be aware of before you run it:

1. Your AUTOEXEC.BAT file (if any) is renamed AUTOEXEC.SAV, and a new one is created that loads your PCTD applications,
2. a subdirectory is created for the PCTD programs and associated files,
3. any copies of FORMAT.COM found on your hard disk are renamed FORMAT!.COM, and
4. a one-line batch file named FORMAT.BAT will be created to run PC FORMAT whenever you wish to format a disk or drive.

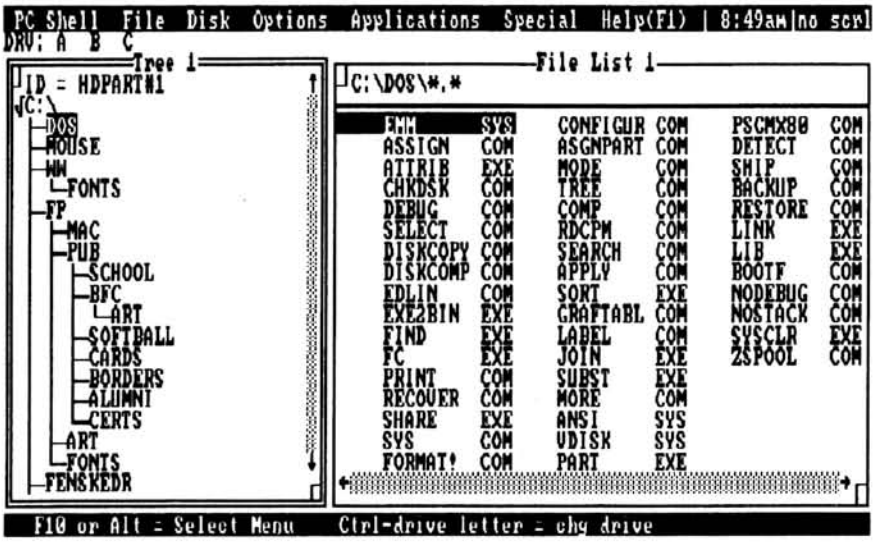
To begin installation, insert Disk 1 into a floppy disk drive, change to that drive, type PCSETUP and press <RETURN>. After verifying the name of the subdirectory, it will store the PCTD files in C:\PCTOOLS (default); the files from Disk 1 are copied to the new subdirectory. You will be prompted to replace the disk with Disks 2, 3 and 4 at the right time, and then the copying is completed. At that point, you will choose which of the major tools will be installed on your system.

You can install PC Shell in resident mode or as your standard DOS shell, in which case, a line will be appended to your AUTOEXEC.BAT file to execute PC Shell as the last batch command. If you prefer resident mode, you need to make a choice between speed of initialization when you call up PC Shell or memory usage. The four options (Tiny, Small, Medium and Large — shades of C programming!!) use from 9K to 170K of memory, but the Large option loads the fastest. If you have at least 170K of expanded (LIM/EMS) memory available, the Tiny option is the best option, since it can load at comparable speed and occupies only a small amount of memory when not in use. I chose Tiny, and the command PCSHELL/RTINY was added to my AUTOEXEC.BAT file.

PC Tools Desktop can be installed in resident mode next. The only decision to make at this point is whether you want to devote 60K of your conventional memory to the BACKTALK program. If you have a modem and you upload or download files from remote systems regularly, BACKTALK can enable those sessions to run in the "background," so that you can use your computer for other tasks during these file transfers. Without installing BACKTALK, the command DESKTOP/R/CS is added to AUTOEXEC.BAT. The Mirror/Rebuild program is installed next, adding the command MIRROR C:/TC to my batch file. Finally, PC-Cache is installed. This program allows data to be read from disks much faster than normal by storing sequential chunks of data in memory so that subsequent reads don't always have to access (slower) disk drives. PC-Cache will occupy 64K of conventional memory unless it can find some unused EMS memory to use instead. The default setting is to cache only reads from the hard disk, so the command PC-CACHE /IA /IB was the final addition to my AUTOEXEC .BAT file. All of these programs have further options you can toggle at startup by editing them onto the appropriate lines in your AUTOEXEC.BAT file.

One final installation note: reboot, and then type the command PATH <RETURN> after you're finished to check your new PATH statement. The PCSETUP program appends the name of the PCTD subdirectory onto the system's PATH statement by editing your AUTOEXEC .BAT file. (If you haven't such a file present, one will be created for you!) If your original PATH was close to the MS-DOS limit of 127 characters, this appended path name may be truncated. PATH statements that exceed this limit can be fixed by either deleting one of your path specifications or shortening it by using the SUBST command to map a specification to a single letter. Make sure your LASTDRIVE statement encompasses this new "drive" letter, or else you'll encounter a wonderfully cryptic MS-DOS error message (see your MS-DOS manual for details on using SUBST and LASTDRIVE)!

After installation of PCTD, I found that I had 514K of conventional memory available and 1.3 megabytes less space on my hard disk. Your mileage will, of course,



```
PC Shell  File  Disk  Options  Applications  Special  Help(F1) | 8:49am|no scrl
DRV: A  B  C
=================Tree 1===========  ----------------File List 1----------------
 ID = HDPART#1                        C:\DOS\*.*
√C:\
  ├DOS                                 EMM       SYS   CONFIGUR COM   PSCMX80  COM
  ├MOUSE                               ASSIGN    COM   ASGNPART COM   DETECT   COM
  ├WW                                  ATTRIB    EXE   MODE     COM   SHIP     COM
  │ └FONTS                             CHKDSK    COM   TREE     COM   BACKUP   COM
  ├FP                                  DEBUG     COM   COMP     COM   RESTORE  COM
  │ ├MAC                               SELECT    COM   RDCPM    COM   LINK     EXE
  │ ├PUB                               DISKCOPY  COM   SEARCH   COM   LIB      EXE
  │ │ ├SCHOOL                          DISKCOMP  COM   APPLY    COM   BOOTF    COM
  │ │ ├BFC                             EDLIN     COM   SORT     EXE   NODEBUG  COM
  │ │ │ └ART                           EXE2BIN   EXE   GRAFTABL COM   NOSTACK  COM
  │ │ ├SOFTBALL                        FIND      EXE   LABEL    COM   SYSCLR   EXE
  │ │ ├CARDS                           FC        EXE   JOIN     EXE   ZSPOOL   COM
  │ │ ├BORDERS                         PRINT     COM   SUBST    EXE
  │ │ ├ALUMNI                          RECOVER   COM   MORE     COM
  │ │ └CERTS                           SHARE     EXE   ANSI     SYS
  │ ├ART                               SYS       COM   VDISK    SYS
  │ └FONTS                             FORMAT!   COM   PART     EXE
  ├FENSKEDR
 F10 or Alt = Select Menu    Ctrl-drive letter = chg drive
```

Figure 1
The Opening PC Shell Screen

vary depending on just how you want to customize installation!

Working With PC Shell

Since the manual describes the workings of all PC Shell commands in great detail, there's no need to cover every command in this walkthrough. I'll select one or two commands from each of the menus and give an example of their use, so that you get a feel for the workings of the PC Shell program. Rather than belaboring all of the keystroke sequences that will work in each case, I'll describe the mouse commands only. With that in mind, let's begin our test-drive of PC Shell!

Start the program by either pressing the Control and Escape keys, or by typing PCSHELL <RETURN> at the DOS prompt. After a few seconds to read system and file information, a two-part display appears on your screen. Notice the tree-oriented subdirectory structure diagrammed on the left-hand side of the screen. One of the subdirectories listed will be highlighted, and the files in that subdirectory are displayed on the right-hand side of the screen. Clicking the mouse on one side or the other will select the "active" side for further selection. The active side is indicated by a double-line border, while the other side has a single-line border. Pulldown menus entitled File, Disk, Options, Applications, Special, and Help(F1) are arranged across the top of the screen. Below that line is a current drive indicator, which you can change with a mouse click if needed (see Figure 1).

Sometimes you'll want to view directory structure for two drives at once. Click on the Options menu, and notice the Two List Display choice. This option will split your screen horizontally, giving you a chance to display a second drive in the lower half. This is quite handy when you want to move files from one drive to another.

While we're exploring the Options menu, note that here is where you can choose different colors, sizes and screen locations for the PC Shell windows. If you don't like the colors or layout chosen by the PCTD programmers, change them! A sample text box will pop up when you change colors, so you can view your artistic talents as you go; another nice touch!

Let's back out of this menu (click the mouse anywhere on the screen outside the Options menu) and explore the File menu next.

File Menu

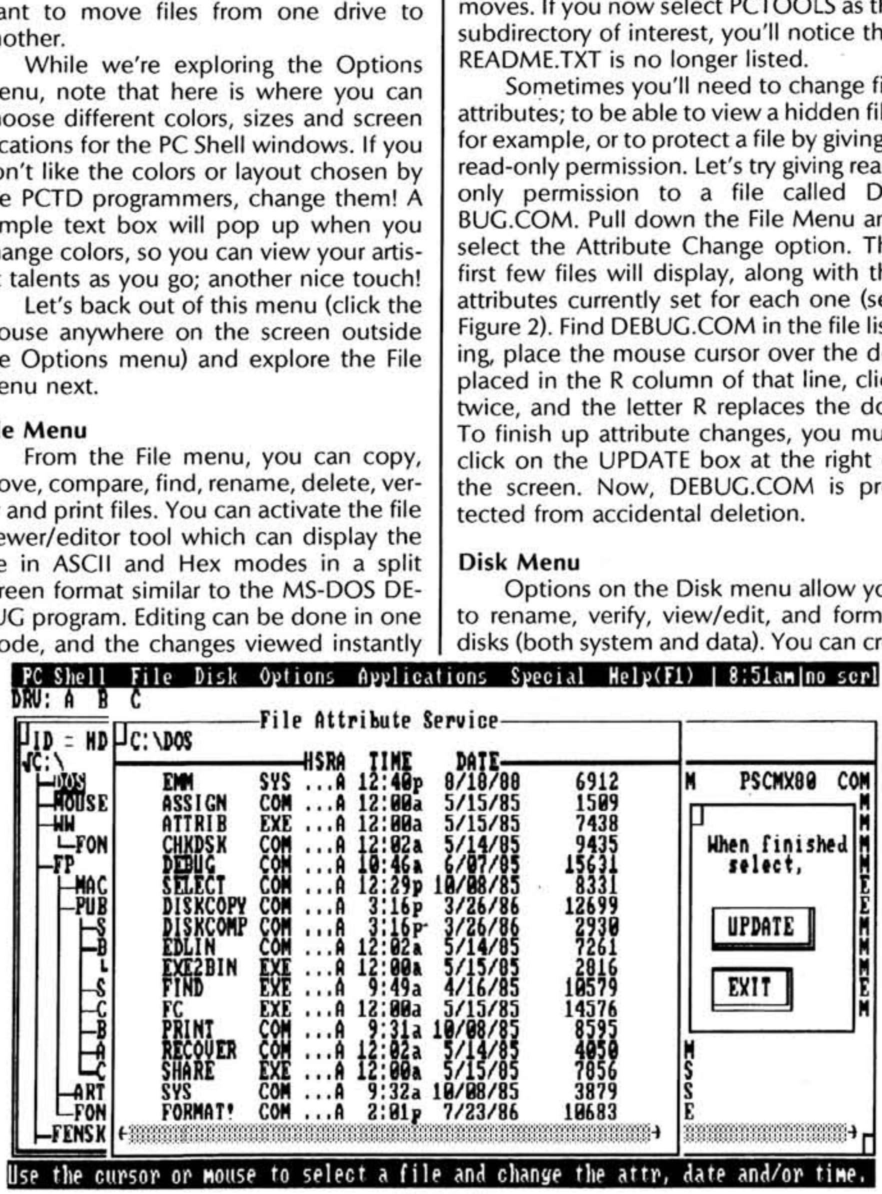
From the File menu, you can copy, move, compare, find, rename, delete, verify and print files. You can activate the file viewer/editor tool which can display the file in ASCII and Hex modes in a split screen format similar to the MS-DOS DEBUG program. Editing can be done in one mode, and the changes viewed instantly in the other mode! In addition, you can change file attributes, get directory or file information, print a directory listing, run executable files, exit to MS-DOS temporarily, or exit the PC Shell program. Let's see what's involved in moving a file and in changing the attributes of a file.

As an example, you might want to move a file called README.TXT from a subdirectory called PCTOOLS to another subdirectory called TEMP. Click on PCTOOLS on the left side of the screen, and the files in that subdirectory appear on the right side. Click on the file README .TXT, pull down the File menu, and select Move. You'll have to click on a CONTINUE box to verify that you want to move the file (the original copy of the file will be deleted). Then, you'll need to select the target drive (from a list provided) and the target subdirectory (back to the left side of the screen!). As soon as TEMP is selected, the screen updates and the file moves. If you now select PCTOOLS as the subdirectory of interest, you'll notice that README.TXT is no longer listed.

Sometimes you'll need to change file attributes; to be able to view a hidden file, for example, or to protect a file by giving it read-only permission. Let's try giving read-only permission to a file called DEBUG.COM. Pull down the File Menu and select the Attribute Change option. The first few files will display, along with the attributes currently set for each one (see Figure 2). Find DEBUG.COM in the file listing, place the mouse cursor over the dot placed in the R column of that line, click twice, and the letter R replaces the dot. To finish up attribute changes, you must click on the UPDATE box at the right of the screen. Now, DEBUG.COM is protected from accidental deletion.

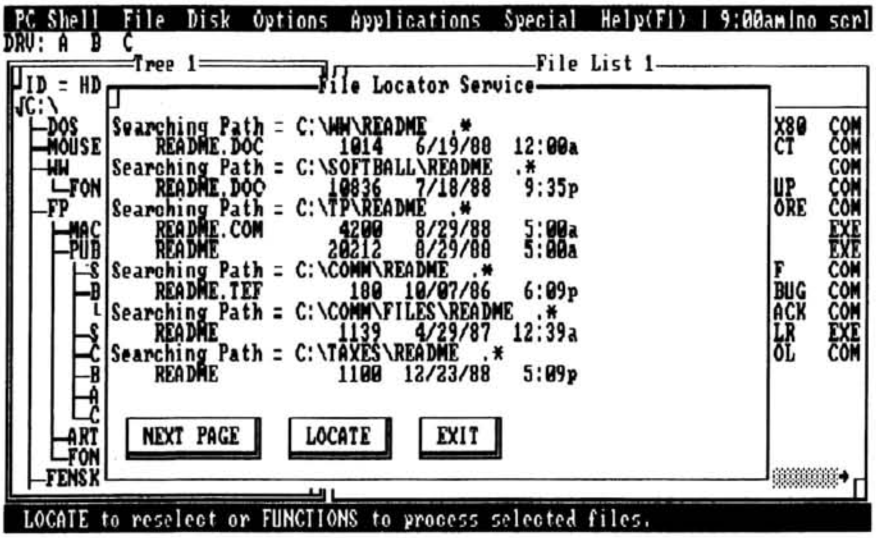
Disk Menu

Options on the Disk menu allow you to rename, verify, view/edit, and format disks (both system and data). You can cre-



```
PC Shell  File  Disk  Options  Applications  Special  Help(F1) | 8:51am|no scrl
DRV: A  B  C
               ---------File Attribute Service-----------------
 ID = HD  C:\DOS
√C:\      -------------------HSRA  TIME     DATE--------------
  ├DOS         EMM       SYS  ...A 12:40p  8/10/88      6912   M   PSCMX80  COM
  ├MOUSE       ASSIGN    COM  ...A 12:00a  5/15/85      1509                  M
  ├WW          ATTRIB    EXE  ...A 12:00a  5/15/85      7438                  M
  │ └FON       CHKDSK    COM  ...A 12:02a  5/14/85      9435    When finished M
  ├FP          DEBUG     COM  ...A 10:46a  6/07/85     15631      select,     M
  │ ├MAC       SELECT    COM  ...A 12:29p 10/08/85      8331                  E
  │ ├PUB       DISKCOPY  COM  ...A  3:16p  3/26/86     12699                  E
  │ │ ├S       DISKCOMP  COM  ...A  3:16p  3/26/86      2930     UPDATE       M
  │ │ ├B       EDLIN     COM  ...A 12:02a  5/14/85      7261                  M
  │ │ │ L      EXE2BIN   EXE  ...A 12:00a  5/15/85      2816                  M
  │ │ ├S       FIND      EXE  ...A  9:49a  4/16/85     10579     EXIT         E
  │ │ ├C       FC        EXE  ...A 12:00a  5/15/85     14576                  M
  │ │ ├B       PRINT     COM  ...A  9:31a 10/08/85      8595
  │ │ ├A       RECOVER   COM  ...A 12:02a  5/14/85      4050   M
  │ │ └C       SHARE     EXE  ...A 12:00a  5/15/85      7056   S
  │ ├ART       SYS       COM  ...A  9:32a 10/08/85      3879   S
  │ └FON       FORMAT!   COM  ...A  2:01p  7/23/86     10683   E
  ├FENSK
Use the cursor or mouse to select a file and change the attr, date and/or time.
```

Figure 2
Charging File Attribute



```
Searching Path = C:\WW\README    .*
    README.DOC       1014   6/19/88  12:00a
Searching Path = C:\SOFTBALL\README   .*
    README.DOC      10836   7/18/88   9:35p
Searching Path = C:\TP\README   .*
    README.COM       4200   8/29/88   5:00a
    README          20212   8/29/88   5:00a
Searching Path = C:\COMM\README   .*
    README.TEF        180  10/07/86   6:09p
Searching Path = C:\COMM\FILES\README   .*
    README           1139   4/29/87  12:39a
Searching Path = C:\TAXES\README   .*
    README           1100  12/23/88   5:09p
```

```
LOCATE to reselect or FUNCTIONS to process selected files.
```

Figure 3
Searching for Occurrences of README.*

ate, remove, rename and move subdirectories using the Directory Maintenance selection. Finally, you can locate files on disks, get further disk information, and park the head of your hard disk to lessen the chance of damage due to the shock of moving your machine. I'll demonstrate file location and subdirectory movement to show some of the capabilities of this menu.

As an example, I searched my hard disk for all files with the name README and any file extension. To do this, I pulled down the Disk menu, selected Locate File, keyed in the file name README, accepted the default extension of *, and clicked on the Locate box. All files matching the search criteria were then listed, along with path name, size, date and time. If there are too many matches to show on one screen, there will be a Next Page box to click for further viewing (see Figure 3). I found 9 such files on my machine, which is a pretty good argument for using subdirectories to separate such things! Remember to click on Exit when you're finished searching for files.

Moving subdirectories is accomplished by choosing Directory Maintenance from the Disk menu, and Prune and Graft from the submenu that appears. I decided to move my TEMP subdirectory under PCTOOLS, so first I was instructed to choose the directory to "prune". Click on TEMP and the Continue box to verify your choice. Then click on PCTOOLS as the destination directory and Continue; the display is updated a few seconds later with the new arrangement. It may not be the answer to ALL of your organization needs, but it certainly makes this kind of restructuring easy.

Application Menu

The Application Menu is used to list programs that you run frequently, along with all of the information PCTD needs to know about them to save you extra keystrokes at runtime. Let's add PC WatchWord (my favorite word processor!) to the menu so I can easily execute it from within PC Shell. Pull down the Options menu and choose Modify Applications List. Click on the New box, and a blank form will appear for you to fill with information about your application. Although I found some of the prompts unclear, the bottom line of the screen displays a useful description of what to enter at the cursor's current position. I keyed in the title (you can prefix one letter of the title with a caret character to set up a "run" character), ran file name, extension, runtime parameters (I use none in this case), and then answered questions regarding details about the runtime environment. Use the hints in the bottom line of the screen, and you'll have no trouble at all filling in the form. Click on Save and then Exit, and a new application joins the list. Each time I want to run PC WatchWord, I pull down the Applications menu, click on my new entry (or type the "run" character, which is highlighted in the title), and I'm in WatchWord.

Special Menu

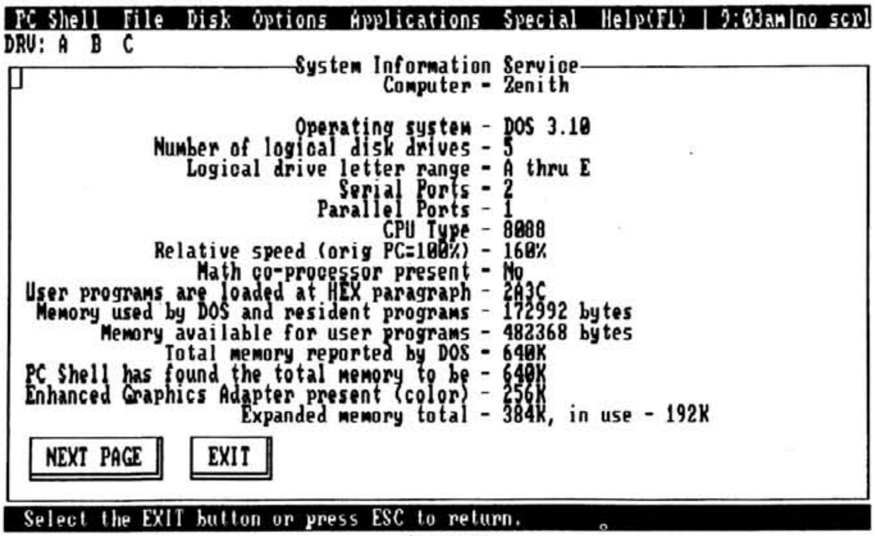
The tools available on the Special menu are some of the best reasons to have PCTD around. You can get detailed information about your computer system, undelete files, map sector usage on files and disks, sort directories, list a memory map showing resident programs currently loaded, and remove PC Shell from memory if you are running in resident mode. Let's explore the System Information and Disk Map options a little further.

When you pull down the Special menu and click on System Information, you'll see a display similar to Figure 4. Your MS-DOS version, number of drives, ports, type of CPU, speed relative to a standard PC and other data will be listed. It's one place to find out for sure how much memory you have on your EGA card if you never knew or can't recall! Click Next Page to continue the listing or Exit when finished.

Back at the menu, if you select Disk Map, you'll see a graphics display of sectors in use, unallocated sectors, and bad clusters on the current drive (see Figure 5). This display gives you a good idea how fragmented your drive is. Another program provided with PCTD (Compress) can help you deal with this problem if you see a display that looks more like a checkerboard than two smooth areas of different design. Click the Exit box to return to the PC Shell main display.

Help Menu

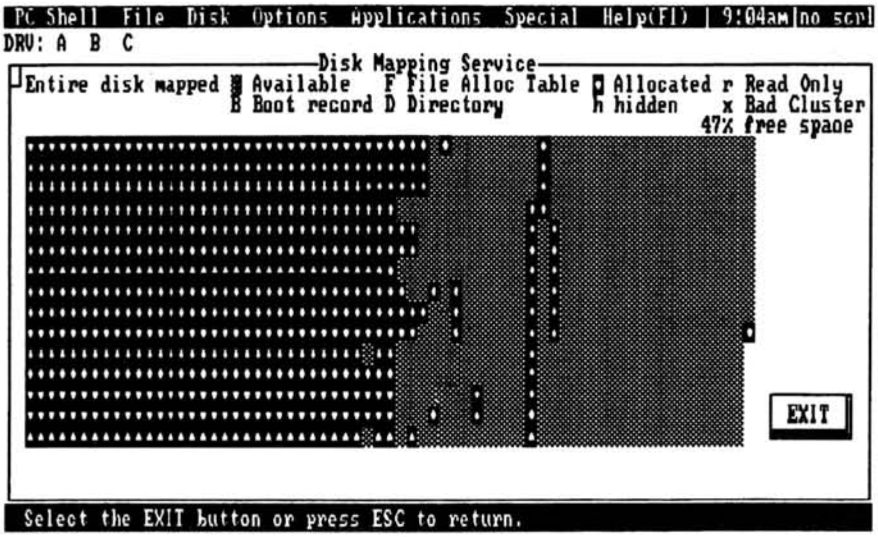
The final menu contains help for those situations where a little advice can clear up temporary confusion. I haven't had to use it very much at all, which says more about the logical organization of PCTD than it does about me!


PC Shell File Disk Options Applications Special Help(F1) | 9:03am|no scrl
DRV: A B C
System Information Service
NEXT PAGE
EXIT

```
                               Computer - Zenith

                       Operating system - DOS 3.10
          Number of logical disk drives - 5
             Logical drive letter range - A thru E
                           Serial Ports - 2
                         Parallel Ports - 1
                               CPU Type - 8088
          Relative speed (orig PC=100%) - 160%
               Math co-processor present - No
User programs are loaded at HEX paragraph - 2A3C
 Memory used by DOS and resident programs - 172992 bytes
      Memory available for user programs - 482368 bytes
            Total memory reported by DOS - 640K
PC Shell has found the total memory to be - 640K
Enhanced Graphics Adapter present (color) - 256K
                   Expanded memory total - 384K, in use - 192K
```

```
Select the EXIT button or press ESC to return.
```

Figure 4
System Information Screen



```
 PC Shell  File  Disk  Options  Applications  Special  Help(F1) | 9:04am|no scrl
DRV: A  B  C
                            Disk Mapping Service
 Entire disk mapped ▓ Available   F File Alloc Table ▢ Allocated r Read Only
                    B Boot record D Directory        h hidden    x Bad Cluster
                                                          47% free spaoe
                                                                      EXIT
 Select the EXIT button or press ESC to return.
```

Figure 5
Hard Disk Map

A Look At PC Format

PC Format is a floppy disk or hard disk formatting program that completely replaces the MS-DOS Format program you received with your system. It can be run as a standalone program by typing FORMAT <RETURN> from the system prompt, because the PCTD installation program renames FORMAT.COM to FORMAT!.COM and creates a FORMAT.BAT file which executes PC Format. You can also format diskettes from within PC Shell using the File menu, or you can run PC Format as an application from within PC Shell.

PC Format works exactly like the MS-DOS Format program you're familiar with, except that a percent complete is continually displayed as the diskette formats. Internally, however, there are some distinct differences between the two format programs. PC Format will first check to see if readable data exists on the disk. Finding some, the program clears the File Allocation Table and moves the first character of every file name into a reserved byte. In this way, the PCTD program Rebuild can recover not only data, but the entire file name if you accidentally format a disk with useful files still on it. In addition to all of the standard MS-DOS Format options, PC Format has a Rewrite option, that will read each track on a diskette, reformat the track, and rewrite the data, cleaning up marginal areas of the disk while leaving the data and files intact. It's everything you've always needed in a Format program, with some extras thrown in to make life simpler!

Until Next Time . . .

In the next installment of this series, I'll describe the capabilities of PC Tools Desktop, a program that compares quite favorably with utilities like Sidekick Plus. I'll also describe how to use PC Secure, a fast data compression, encryption and decryption tool and I'll perform some "time trials" with PC Cache, to see just how fast disk accesses can be with the cache in place. The final installment will cover PC Backup, a better backup utility than DOS offers, along with the hard disk programs Compress and Mirror/Rebuild. There have been some enhancements in the latest release (Version 5.1) of PCTD, and I should get my version upgraded in time to cover this latest release in the next two installments.

If you have any questions about PC Tools Deluxe Version 5 or anything mentioned in this article, please feel free to drop me a line. I am not connected in any way with Central Point Software, except as a satisfied user of two of their products. See you next time!!

Products Discussed

PC Tools Deluxe Version 5
Central Point Software, Inc.
15220 N.W. Greenbrier Pkwy
Suite 200
Beaverton, OR 97006-5764

*

*** 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/calander. $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 or 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. EasyI/O-1, One Serial Port $91.00. EasyI/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 compatable or special controller only.

** SEAGATE HARD DISK DRIVES ***

ST-125	21 MEG, 28 MS, Auto Park Heads	$ 275.00
	With Controller & Cables	$ 329.00
ST-138	31 MEG, 28 MS, Auto Park Heads	$ 329.00
	With Controller & Cables	$ 383.00
ST-238	31 MEG, 65 MS, RLL	$ 258.00
	With RLL Controller & Cables	$ 309.00
ST-251	42 MEG, 40 MS, Auto Park, Software	$ 384.00
	With Controller & Cables	$ 438.00
ST-251-1	42 MEG, 28 MS, Auto Park, Software	$ 465.00
	With Controller & Cables	$ 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, Keytronics 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/calander, 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 RRL hard disk drive $1126.00

Example #3
IBM AT Compatible 80286 12 Meg CPU, 2 Meg RAM, clock/calander,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

MasterCard

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

VISA

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.

Reader Service #117

SKY

Larry Bollman
Heath/Zenith Computer Consultant
Heath/Zenith, St. Joseph, MI

"What's that star, Daddy?" said my daughter pointing to the brightest light in the sky. Sound familiar? Being an amateur astronomer, I usually could be sure they were pointing at a planet, but some children would point at a star. If they were looking at a planet, I was confident enough to tell them that it wasn't a star and name the planet. However, when they pointed at a star, they would accept the fact that Daddy wasn't a "know-it-all", or laugh it off when I'd say "That's Fred."

Then I got "The_Sky". It's a software program that every amateur astronomer should have if they have a Zenith MS-DOS PC compatible computer. (Who's IBM?)

Being a Heath/Zenith Computer Consultant, software sometimes just appears on my desk. The_Sky appeared just before Halley's comet made its visit, so it was a welcome addition. The only thing I didn't like was there was no reference to who wrote the program or who I could suggest customer's to contact in the program. I finally saw an advertisement in Sky and Telescope's July issue, and contacted them for information.

The first thing I suggested was to put their company name and phone number in the program. The girl was very helpful, and I informed her of the fact that I couldn't get the program to work on the AT's or the newer H-386 computers. She said that they had received the same complaint and a newer version was available.

Besides being very easy to use and an excellent screen representation of the night (or day) sky, the program is a good one to test the operation of a math coprocessor chip. Without the math chip in a PC, it takes almost a minute and a half to do its number crunching. It draws a line across the screen as it computes. With a math chip, it takes five seconds or less.

In the mean time, I sold my telescope, thinking of getting something lighter to carry around. So far I'm still thinking.

The new version arrived and I don't know how they did it, but it's even better! It seems that these people listen to what their customers want. Their company name and address was already in the program. Of course, they did it just for me! Now all they need to do is add it to their manual and include the phone number.

After booting and renaming the program to SKY.EXE, the program was run. (I hate to type more than necessary.) From the introduction, the setup menu was entered just by hitting the RETURN key.

Once in the setup menu, you can input several locations on Earth using Latitude and Longitude. I used the Heath portable Loran C locator, MI-3000, to get the exact position of my house. 42.30.00 Latitude and 86.00.35 Longitude. They also list several cities in their manual, so your location can be approximated.

The time zone was entered as 5, although I'm closer to 6, but that can just as easily be changed if things don't look right.

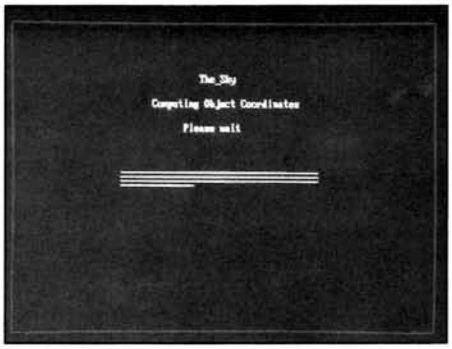
It supports four levels of video. Two levels of CGA and two of EGA.

(1) 320 × 200 CGA four colors.
(2) 640 × 200 CGA two colors.
(3) 640 × 200 EGA sixteen colors.
(4) 640 × 350 EGA sixteen colors.

I changed the arrow speed to their lowest suggested number, and it wouldn't take it. The manual stated 0 - 10, but it should be 1 - 10. Entering a 1 allowed me to continue. For the H-386, a 1 was still too fast, and I let them know of these minor problems already. I later learned how to zoom in a variable area of the sky, then use of the arrow pointer was easier.

You can either load the Big.sky data (default), enter Small.sky (which uses a smaller data file), or enter a modified file that you can create with their sky_edit program. Why anyone would want to use small.sky, I don't know. I did it just to see how many fewer stars it would load. For the first run, I used the Big.sky setting.

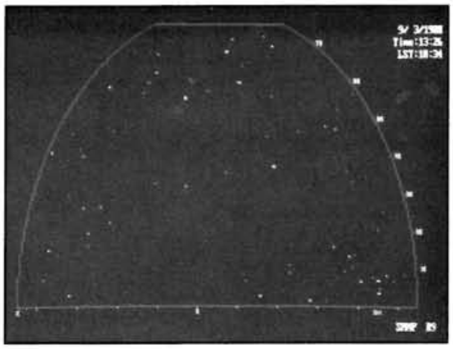
After everything was set properly, I typed the F10 key to save the settings and use them next time the program was run. The ESC key is used to exit and you have to remember this on some screens, but not this one.

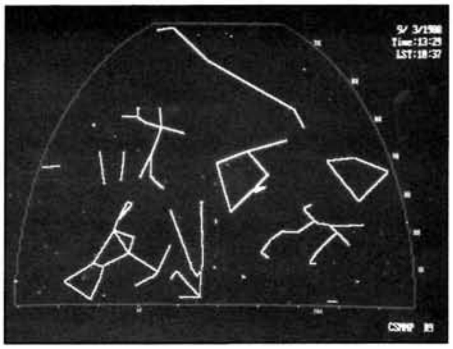


```
The_Sky

Computing Object Coordinates

Please wait
```

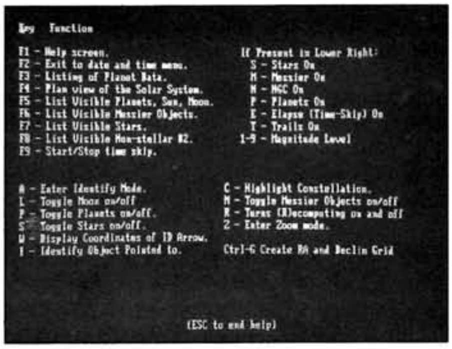
Hitting the spacebar causes it to load 993 stars, 109 messier objects, and 9 non-stellar (Planets). It then proceeds to draw three and three quarter lines across the screen as it computes the data. The time depends on the computer speed and whether a math chip is installed as explained above and in their manual.

If your star isn't shown, it can be added, and the manual explains how to use the Sky_edit program to do this, and save under a name of your choosing.

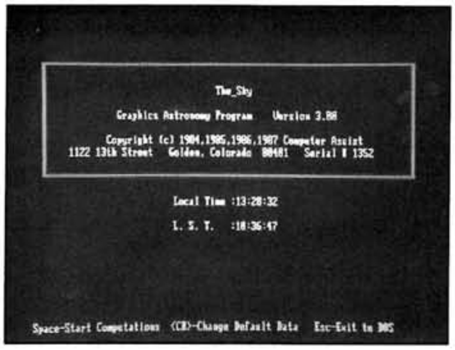




Once the sky display is shown, they call this the Scroll Mode Display, you are in the "base mode". All the function keys branch off from this display, and you must return, using the ESC key, to go to another. The arrow keys move the objects around the screen as if you were turning East or West. Using the arrow keys, either left or right, make sure your seat belt is fastened! The screen display effect is impressive. If you want, you can face the East and watch the stars rise. Or vice versa, face the West and watch them set.

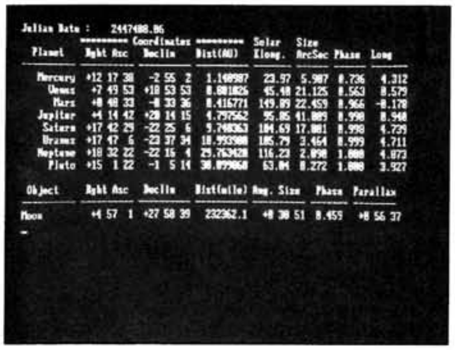


```
Key  Function

F1 - Help screen.                          If Present in Lower Right:
F2 - Exit to date and time menu.             S - Stars On
F3 - Listing of Planet Data.                 M - Messier On
F4 - Plan view of the Solar System.          N - NGC On
F5 - List Visible Planets, Sun, Moon.        P - Planets On
F6 - List Visible Messier Objects.           E - Elapse (Time-Skip) On
F7 - List Visible Stars.                     T - Trails On
F8 - List Visible Non-stellar #2.          1-9 - Magnitude Level
F9 - Start/Stop time skip.


A - Enter Identify Mode.                C - Highlight Constellation.
L - Toggle Moon on/off                  M - Toggle Messier Objects on/off
P - Toggle Planets on/off.              R - Turns (R)ecomputing on and off
S - Toggle Stars on/off.                Z - Enter Zoom mode.
W - Display Coordinates of ID Arrow.
I - Identify Object Pointed to.         Ctrl-G Create RA and Declin Grid



                        (ESC to end help)
```



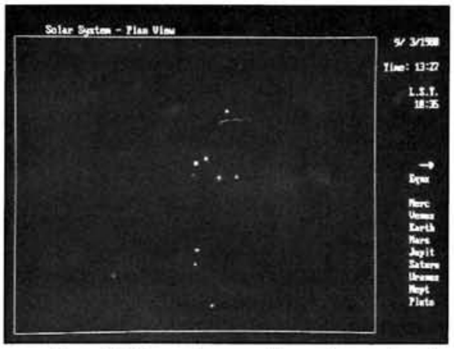
```
                    The_Sky

  Graphics Astronomy Program    Version 3.00

  Copyright (c) 1984,1985,1986,1987 Computer Assist
1122 13th Street   Golden, Colorado  80401   Serial # 1352

              Local Time :13:28:32

              L. S. T.   :18:36:47

Space-Start Computations  <CR>-Change Default Data   Esc-Exit to DOS
```

Hitting the F1 key displays all the options available. Avoid F2, as this will take you back to the start of the program. Without a math coprocessor you may feel the urge to curse.



```
Julian Date :    2447408.06
         ========= Coordinates =========  Solar   Size
 Planet   Rght Asc   Declin    Dist(AU)   Elong.  ArcSec Phase  Long

 Mercury +12 17 30   -2 55  2  1.1409987   23.97   5.987 0.736   4.312
   Venus  +7 49 53  +18 53 53  0.681026    45.40  21.125 0.563   0.579
    Mars  +0 40 33   -0 33 36  0.416771   149.09  22.459 0.966  -0.170
 Jupiter  +4 14 42  +20 14 15  4.797562    95.05  41.009 0.990   0.940
  Saturn +17 42 29  -22 25  6  9.740363   104.69  17.001 0.998   4.739
  Uranus +17 47  6  -23 37 34 18.993900   105.79   3.464 0.999   4.711
 Neptune +18 32 22  -22 15  4 29.763420   116.23   2.090 1.000   4.873
   Pluto +15  1 22   -1  5 14 30.099060    63.04   0.272 1.000   3.927

Object    Rght Asc   Declin   Dist(mile) Ang. Size  Phase  Parallax

Moon       +4 57  1 +27 58 39  232362.1  +0 30 51  0.459  +0 56 37
-
```

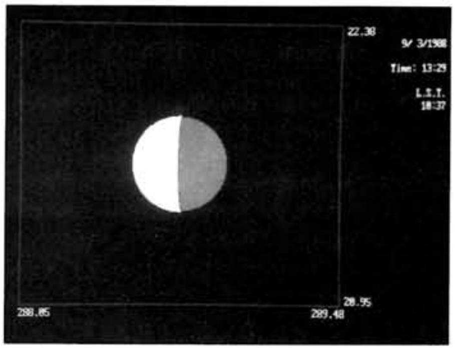
The F3 key is not explained in the manual, but pressing this key produces a listing of the planets and moon data. Such as Right Ascension, Declination, Distance, and other astronomy type information. For those with a good telescope and properly aligned mount, this will aid in finding the planets.



```
  Solar System - Plan View
                                          9/ 3/1988

                                         Time: 13:27

                                             L.S.T.
                                              18:35

                                                 →
                                               Dyna

                                               Merc
                                               Venus
                                               Earth
                                               Mars
                                               Jupit
                                               Saturn
                                               Uranus
                                               Nept
                                               Pluto
```

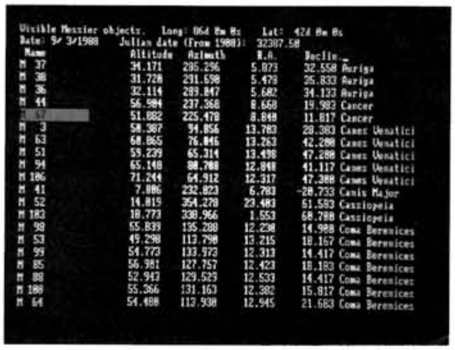
The F4 key shows the nine planets and their relative positions looking down from space on their orbits. This makes it easier for the children to understand why Venus looks closer to the sun, at times, than Mercury.

The F5 key produces a listing of the planets, sun and moon if they are available. You can highlight any one, hit the return key and the sky view will show the area the object selected should be in on the main screen. Typing the 'A' key will produce an arrow pointing at the object. One neat thing I learned from the manual, is that the selected object can be zoomed in by typing certain keys. Since my intention is to make you aware of the product, and not act as a substitute for their manual, not all features will be discussed.



```
                                   22.38
                                              9/ 3/1988

                                             Time: 13:29

                                                 L.S.T.
                                                  18:37

                                  20.95
288.05                        289.48
```

I zoomed in on all the planets and observed the shadow produced by their position in relation with the sun. I was hoping to see Saturn's rings, but that was expecting too much. Maybe in a future version they can add more detail (Hint, hint).



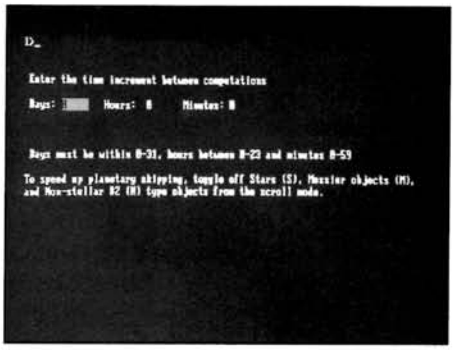
```
Visible Messier objects.   Long: 086d 0m 0s    Lat: 42d 0m 0s
Date: 9/ 3/1988    Julian date (From 1900): 32387.50
 Name              Altitude   Azimuth     R.A.     Declin._
M  37               34.171    285.296     5.873    32.550 Auriga
M  38               31.720    291.690     5.470    35.833 Auriga
M  36               32.114    289.047     5.602    34.133 Auriga
M  44               56.904    237.368     8.660    19.983 Cancer
M  67               51.002    225.470     8.840    11.017 Cancer
M   3               58.387     94.056    13.703    28.383 Canes Venatici
M  63               60.865     76.046    13.263    42.200 Canes Venatici
M  51               59.239     65.314    13.490    47.200 Canes Venatici
M  94               65.140     80.700    12.840    41.117 Canes Venatici
M 106               71.244     64.912    12.317    47.300 Canes Venatici
M  41                7.006    232.823     6.783   -20.733 Canis Major
M  52               14.819    354.270    23.403    61.583 Cassiopeia
M 103               10.773    330.966     1.553    60.700 Cassiopeia
M  98               55.839    135.200    12.230    14.900 Coma Berenices
M  53               49.290    113.790    13.215    18.167 Coma Berenices
M  99               54.773    133.973    12.313    14.417 Coma Berenices
M  85               56.981    127.757    12.423    18.183 Coma Berenices
M  88               52.943    129.529    12.533    14.417 Coma Berenices
M 100               55.366    131.163    12.382    15.817 Coma Berenices
M  64               54.400    113.930    12.945    21.683 Coma Berenices
```

The F6 key produces a listing of all the Messier Objects. The same applies here in that the object can be selected and viewed expanded. I ran into trouble here when I kept zooming in until it produced negative numbers. At this point, the objects look like an approaching galaxy. That messed-up the data and I got garbage in the F5 and F6 screens. Re-loading the program was the only way to restore proper operation. This problem occurred on three different computers, an H-2526 and two H-386s. There is no more detail when zooming in on these objects. I tried Andromeda Galaxy, M-31, which is the easiest to find, and it remained a dot until I exceeded the zoom feature.

The F7 key produces a listing of the stars and F8 lists the non-stellar objects. These can be found and viewed the same way as above.

The F9 key is for setting the time interval between computations.

All the objects can be turned on and off with the touch of a button, such as 'S' for stars, 'M' for Messier objects, 'P' for Planets, and 'N' for the NGC objects.



```
D_

 Enter the time increment between computations

 Days:       Hours: 0     Minutes: 0


 Days must be within 0-31, hours between 0-23 and minutes 0-59

To speed up planetary skipping, toggle off Stars (S), Messier objects (M),
and Non-stellar #2 (N) type objects from the scroll mode.
```

The 'T' key produces a 'trailing' function that leaves a trail of everything that was left on. A math chip is a must for this function, but if you don't mind waiting, you can set the screen update for a future date. Then with the trail function on, see how much the position of the objects change.

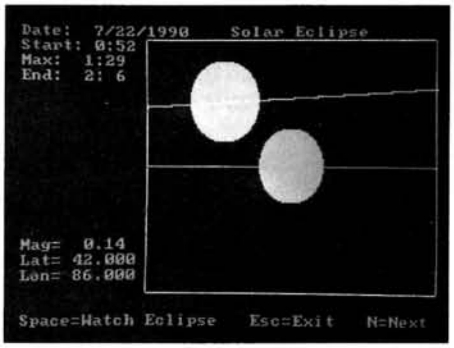
The 'R' key will turn off re-computing so that the screen will not change. The 'E' key turns on the Elapse mode. This is used in conjunction with the F9 key.

The number keys, 1-9 is used to set the magnitude level of the stars which will be shown on the screen.

Their example of the sun's analemma is to position yourself at about 60 degrees latitude, turn everything off, except the sun, enter a time increment of 10 days, and press the 'T' key. This will produce the figure eight on the screen. A superimposed photograph taken at noon, over a years period, is the only way I have seen this duplicated.

I have seen other sky programs, and none have compared to 'The_Sky'. It comes without copy protection (three cheers!) and sells for $60.00. With it comes their other program called Sky_Calc. This one is already abbreviated to SC.EXE, so I didn't have to use rename here.

Sky_Calc is another great program! It is intended to compliment The_Sky, and performs common astronomical calculations with little effort. Just two of the features I like the most are its views of Jupiter and its moons, and Lunar and Solar eclipses.



```
Date:   7/22/1990     Solar Eclipse
Start: 0:52
Max:   1:29
End:   2: 6




Mag=  0.14
Lat= 42.000
Lon= 86.000

Space=Watch Eclipse    Esc=Exit    N=Next
```

After typing SC and ENTER, the main menu is displayed. It's like a LOTUS menu in that you either move the highlight cursor or type the first character of the submenu. The selections are File, Positional, Conversions, Miscellaneous, and Options.

In File, you enter the time and date,

Continued on Page 32

POWERING UP

Other Useful DOS Commands

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

The DOS commands introduced in previous articles have been specifically related to that particular subject, and they have been essential to the effective use of that DOS feature or capability (e.g. subdirectories or I/O redirection). In this article, we will look at some of the more advanced commands and DOS features that you can use to help solve a problem that you may encounter.

Since the last article discussed setting up and using a hard disk, we will begin with some commands that can help you in utilizing it. If you have a hard disk, you will probably find that, at some point, you cannot find a file name that you are absolutely certain is somewhere on the disk. It may be in a third or fourth level subdirectory, and it is a real chore to step through each subdirectory with the DIR command in an attempt to locate a "missing" file. There is a much easier way.

The SEARCH Command

SEARCH is an external command that will help you locate files within the subdirectory structure on a disk, and it is included with most versions of Zenith MS-DOS. The syntax for the command is shown in Figure 1.

```
SEARCH [d:][path]afn[/C/T/D]
```

Figure 1
SEARCH Command Syntax

The SEARCH command line must include an ambiguous file name (afn) that may contain either the question mark (?) and/or asterisk (*) wildcards. The optional /C switch will search the Current directory only. The optional /D switch will display the Directory names as they are being searched. The optional /T switch will display the Tree structure similar to the TREE command which we will look at next.

Because the SEARCH command will only search the current directory and all subdirectories below it, it is important to be sure you are in the root directory when you enter the command. Then, you can look for a specific file with the following command:

```
C:\ ==>SEARCH COMMAND.COM
```

SEARCH will display the complete path with the file name COMMAND.COM so you can see exactly where that file is. If you want to find out where all SuperCalc data files are, you could search for all CAL file types with:

```
C:\ ==>SEARCH *.CAL ¦ MORE
```

Again, you will get a display showing all CAL files, and command piping is used with the MORE command in case there are more than 23 files. If you have a hard disk, you will probably find any number of uses for the Zenith SEARCH command to help you locate file names. Although you can use the SEARCH command to display your directory structure with the optional /T switch, you can also use the TREE command that is pretty much standard in most current DOS versions.

The TREE Command

TREE is an external DOS command that will display directory paths and optionally list the file names in each path. It is used to take a quick look at a disk's directory structure. The command syntax is shown in Figure 2 below.

```
TREE [d:][/F]
```

Figure 2
TREE Command Syntax

TREE will display a list of subdirectories on the current drive or the specified drive if the optional drive letter *d* is entered. The optional /F (Files) switch will list the files in each subdirectory. The display does not include the root directory unless the /F switch is entered to include the file names in each directory. The TREE command is included in most Zenith PC compatible DOS versions although it is not available in the Z-100 version.

Sometimes it is useful to be able to easily copy all files in a directory and all of its child subdirectories to another subdirectory on a disk. This is one way to effectively "rename" a subdirectory, and current DOS versions have a command that can help you perform these functions.

The XCOPY Command

Unlike COPY, XCOPY is an external command (note the X) that copies all specified files and/or entire subdirectories from a specified drive and/or path to another. XCOPY has many of the same features as the internal COPY command, and the command syntax is shown in Figure 3.

While that syntax may look somewhat intimidating, it is generally the same as the COPY command and has the same general form of:

```
XCOPY from-location to-location
```

Drive letters, paths, and file names are specified in the same way as COPY, and wildcards may be used in the file name specifications.

The most obvious advantage of the XCOPY command is its capability to copy all files from a specified directory and all

```
XCOPY [from-d:][from-path]from-afn [to-d:][to-path]to-afn[/p]
XCOPY [from-d:][from-path]from-afn to-d:[to-path][/p]
```

Figure 3
XCOPY Command Syntax

its child directories to another location. For example, you could create a backup copy of a disk containing subdirectories with the command:

```
A:\==>XCOPY *.* B:\/S
```

This command would begin the copy from the current drive in the default directory (shown as the root directory on drive A here) and copy all files (*.*) beginning at the root directory on drive B (B:\). Because the /S switch was also entered, XCOPY will also copy all files from all subdirectories on drive A to drive B. If the subdirectory name does not exist on drive B (and it probably will not), XCOPY will create it. In addition to the /S switch, there are a number of other useful switches in this command.

The /E (Empty subdirectories) switch will create subdirectories on the destination even if they are empty and may only be used when the /S switch is also entered. The /A (Archive) switch copies files that have the archive bit set, but *does not* reset the archive bit. The /M (Marked with archive bit) switch also copies files that have the archive bit set, but it *does* reset the archive bit. The /D:mm/dd/yy (Date) switch copies files whose date is equal to or later than the date specified. These are just a few of the optional switches that may be used with XCOPY, and you may want to check your DOS manual for additional ones.

Most of the commands discussed thus far have been more useful for a hard disk system even though they can also be used on floppy disk systems. Now let's take a look at a couple of commands that can be useful on either.

Comparing Files

Regardless of whether you have a hard disk or not, sometimes you are not sure exactly how one file is different from another. You want to compare two different files to see how they are different or perhaps you want to check a current file against a backup copy to see how they are different. There are two DOS commands that can help you do this.

COMP is an external command that compares two files and displays their differences (if any). This command is only available on PC compatible computers, and it is not available with the Z-100 MS-DOS. Refer to Figure 4 for the command syntax.

```
COMP [d:][path]ufn-1 [d:][path]ufn-2
```

Figure 4
COMP Command Syntax

The COMP command compares two unambiguous file names — ufn-1 and ufn-2 — and either or both may be preceded by an optional drive letter and path specification. Although you can use wildcards to specify ambiguous file names for the COMP command, that may not compare the specific files you wanted, so I recommend always using the ufn specifications as shown.

COMP can also be used in the "prompt" mode as shown on the second line, and the command will request both file names with prompts similar to:

```
Enter primary file name:
Enter 2nd file name or drive id:
```

If the specified files are identical, you will see a message like "Files compare ok" when the comparison is completed. If the files have different sizes as listed in the disk directory, COMP assumes that the files are different and terminates with a "Files are different sizes" message. If the files have the same size and are different, COMP displays a message showing the location of the differences with a message like:

```
Compare error at OFFSET hhhhhhhh
File1 = hh
File2 = hh
```

The program identifies the location of the difference (the OFFSET) at a hexadecimal location (hhhhhhhh). It also lists the hexadecimal value (hh) of the bytes that are different in each file so you can see what the difference is. Unless you really want to spend some time learning about the hexadecimal numbering system, the COMP command is not particularly useful. In addition, it is usually difficult to locate the difference(s) in a file (using a word processor for example) because the location is displayed in hex. You won't find too many word processors or other applications that will help you pinpoint a place in a file indicated by a hex number. Fortunately, most versions of Zenith MS-DOS include another command that is significantly better.

FC (File Compare) is also an external command that also allows you to compare two files, and it is available in most Zenith MS-DOS versions (including the Z-100). It has a similar syntax as shown in Figure 5.

```
FC[/p1/p2/p...] [d:][path]source1-ufn [d:][path]source2-ufn
```

Figure 5
FC Command Syntax

FC compares two unambiguous file names (no wildcards) that both may have optional drive and/or path specifications. Note that all switches for FC must be entered immediately following the command which is different from most other DOS commands. For some strange reason, switches used with the FC command *must* be in lowercase letters in most Zenith MS-DOS versions.

FC is much more flexible than COMP. FC can be used to compare both ASCII and binary files. If the file type is COM, EXE or OBJ, FC assumes a binary file comparison; otherwise, an ASCII comparison is assumed. Remember that a CTRL-Z is used to indicate an End-of-file (EOF) for an ASCII file so that the comparison stops when the program finds the first CTRL-Z character. In a binary file, there can be (and usually are) LOTS of CTRL-Z characters.

The FC command has a number of optional switches that can be used in comparing files, and all switches should be entered in lower case letters. Since the availability and functions of some of these switches depends on which version of Zenith MS-DOS you have, you should refer to your manual for additional information on these switches.

The Virtual Disk Revisited

As you may recall, a virtual disk is basically a drive letter that is associated with an "imaginary" disk drive — that is, anything that "looks" like a single physical hard disk drive or partition, or floppy drive, but really is not. Although the term "virtual disk" is normally associated with the VDISK device driver that we have already discussed, there are several DOS commands that can be used with a virtual drive letter.

Among the more important of these commands is the SUBST command. With this command, you can specify a virtual drive letter to effectively "rename" a subdirectory. This can be especially important to you if you are using older software that is not capable of working with subdirectories. But if you use any of the commands that assign virtual drive letters, you will probably need to update the LASTDRIVE= command in the CONFIG.SYS file.

The usual DOS default is that drive letter E is the "maximum" or last drive letter that can be used for additional virtual or non-physical drives. This helps keep the disk drive "table" stored in memory short to maintain best system performance. Even though most of the latest DOS versions allow 26 drive letters (A-Z), you are nearly always safe if you restrict the "last" drive letter to P (i.e.

LASTDRIVE=P) in every DOS version that has been released, and even the older software should be compatible with that standard. The drive letter P is quite specific because early DOS versions (e.g. version 1) only allowed up to 16 drive letters (i.e. A through P) to be used for drive assignments. In most cases, you can use the LASTDRIVE=Z with relative safety, but you will have to check out your software to make sure that there are no problems if you decide to do this. Just be sure that you update the CONFIG.SYS file with the appropriate LASTDRIVE= command if you expect to use any virtual drive letter greater than E with the SUBST command; otherwise, you will see an "Invalid parameter" error message.

The SUBST command

SUBST is an external DOS command that substitutes a disk drive letter for a complete path specification (including a disk drive letter). After using the SUBST command, you can "change" to a new subdirectory by entering a drive letter followed by the usual colon (e.g. W:). As previously mentioned, this command will help you use path specifications with older software that does not have the capability to use subdirectory names. Its syntax is shown in Figure 6.

```
SUBST subst-d: [old-d:]\path (Substitute drive)
SUBST (Display status)
SUBST subst-d: /D (Disconnect drive)
```

Figure 6
SUBST Command Syntax

In the first command form, you can substitute a new (or virtual) drive letter *subst-d* for a complete *\path* specification including the old drive letter *old-d* for that path. The backslash (\) is shown as part of the path because the path specification MUST begin at the root directory for the SUBST command. Although the *old-d* drive letter is optional, it is good practice to enter it for this command. After you have executed the SUBST command to rename a subdirectory with a substituted drive letter (i.e. subst-d), you can log on to that subdirectory by entering the drive letter followed by a colon (:) in the usual way.

In case you forget which subdirectories have been assigned to what drive letters, you can use the SUBST command by itself to display the substituted drive letters and their corresponding paths. And the third command form uses the /D (Disconnect) switch to disconnect the substituted drive from the path. In some DOS versions, you must type a space between the drive letter and the /D switch as shown in Figure 6. Let's see how you might actually use this command and its various forms.

Assume you have a subdirectory for WordStar documents (\WSDATA) on drive C. In order to use an older WordStar version with this subdirectory, you could use the SUBST command to define the \WSDATA subdirectory as drive W by using:

```
C:\ ==>SUBST W: C:\WSDATA
```

Then you can use WordStar to log onto drive W to access the WordStar documents without fiddling with the CHDIR (CD) command to change subdirectories. In this example, you could also enter the SUBST command by itself to display the substituted drive as:

```
W: => C:\WSDATA
```

You can also disconnect the substituted drive W from its defined path by entering the command:

```
C:\ ==>SUBST W: /D
```

Remember that drive W is not a physical disk drive, and you MUST be sure to add the LASTDRIVE=Z command to the CONFIG.SYS file before you can use the drive letter.

Although you can use an existing physical drive letter for the subst-d if you wish, that is not recommended because you will not be able to access that physical drive as long as the SUBST command is in effect for that drive letter. In addition, you cannot use the default drive letter as the *subst-d*, although you may use the current drive letter in the *old-d* specification. Similarly, you cannot use the current directory path as the *\path* in the command line. Now let's see how to JOIN a drive letter to a subdirectory name.

THE JOIN COMMAND

JOIN is an external DOS command that joins a drive letter to a path specification that may also include an optional drive letter. After using the JOIN command, you can effectively change to a new disk drive letter by using the Change Directory (CHDIR or CD) command or use that path in a command line. The JOIN command can be used to effectively create what looks like one single disk drive as far as DOS is concerned. Like the SUBST command, the JOIN command has three syntax forms as shown in Figure 7.

```
JOIN join-d: [old-d:]\new-dir     (Substitute drive)
JOIN                              (Display status)
JOIN join-d: /D                   (Disconnect drive)
```

Figure 7
JOIN Command Syntax

In the first command form, you can join a drive letter *join-d* with a new directory name *new-dir* and optional drive letter *old-d* for that directory. The backslash (\) is shown as part of the directory specification because the path (i.e. new-dir) MUST begin at the root directory. Also, the JOIN command only allows (and requires) one directory level so that the root directory (indicated by the \) is always the parent directory. In other words, the *new-dir* must be in the form of "\YOURDIR" or "\MYDIR"; it cannot be something like "\YOURDIR\MYDIR." In addition, the JOIN command requires that the new directory name *new-dir* be completely empty; *new-dir* cannot contain any files because it is used exclusively by the JOIN command. If the new directory name *new-dir* does not exist, the JOIN command will automatically create it.

Although the *old-d* drive letter is optional, it is good practice to enter it as previously discussed in the SUBST command. After you have executed the JOIN command, you can "log on" to a different disk drive with the CHDIR or CD command.

Like the SUBST command, the JOIN command by itself displays the joined drive letters and their new directory names. And the third command form uses the /D (Disconnect) switch to disconnect the joined drive from the new directory. In some DOS versions, the space between the drive letter and the /D switch is required as shown in Figure 7. Let's see how you might actually use this command and its various forms.

Assume you have a floppy disk system with two drives: A and B. You could then effectively create a single drive system by entering:

```
A:\ ==>JOIN B: A:\DRIVEB
```

The new directory names DRIVEB was selected because it is descriptive of which drive is accessed through its corresponding subdirectory name. Now you can "log on" to drive B by entering the CD \DRIVEB command.

If you use the JOIN command to join drive B to a subdirectory name, you may want to see which drive letters have been used for what directory names. Just enter the JOIN command by itself, and, in this example, you would see:

```
B: => A:\DRIVEB
```

Note that the JOIN display is identical to that used by the SUBST command which is one reason for my particular choice of the \DRIVEB subdirectory name. I recommended you use this convention so it will be obvious what directory name(s) have been used with the JOIN command. And if you want to disconnect the drive letters from the directory names, you can enter the following command:

```
A:\ ==>JOIN B: /D
```

JOIN also has some specific requirements for its use. The joined drive letter *join-d* must be a valid and active drive letter in the system (i.e. A or B or a hard disk partition) or it may be a memory disk created with the VDISK.SYS device driver. Also, the joined drive letter *join-d* cannot be the current drive, and the \new-dir cannot be the current directory. After the JOIN command is executed, you cannot access the joined drive *join-d* by its drive letter — you must use the new directory name ^*new-dir* until it's disconnected by the /D switch.

If the new directory name \new-dir does not exist, JOIN will try to create it; but if \new-dir does exist, it must be empty. Any attempt to JOIN a drive with a directory name that contains files will result in a command failure with the display of the "Directory not empty" error message. After you have disconnected the joined drive from its directory name, you can remove the directory as usual with the RMDIR or RD command if you wish.

You may find the SUBST and JOIN commands will help you solve a number of unusual problems, but you may find a number of more common problems.

The DOS Environment Space

The ENVIRONMENT SPACE is a reserved area of memory used by the Command Interpreter COMMAND.COM to store information that is used by DOS and application programs. In particular, the environment space is specifically used to store the exact location (i.e. path) of COMMAND.COM so that the transient portion can be reloaded when necessary. In addition, the environment space also stores the default PATH= value which is a null path (i.e. "No path") although it can be changed with the PATH command. If you have used the PROMPT command, that value is also stored in the environment space. You can also display or change the contents of the environment space with the SET command as you will see in the next section.

All DOS versions have had an environment space. When DOS version 2.00 was released, the user could access the environment space with the SET, PATH, and PROMPT commands. In most version 2 DOS releases however, the size of the environment space was fixed, and there was no way to change its size. Unfortunately, this size limitation complicated the use of some programs because there are occasions when you need to a number of SET commands (e.g. some programming compilers require this) in addition to the PROMPT and PATH commands.

Although the size limitation problem was essentially fixed in the release of DOS version 3, it was not very well documented and is not very well understood. You can actually change the size of the environment space using the Command Interpreter, but let's see what the environment space actually contains with the SET command.

The SET Command

SET is an internal DOS command that can display current values, or insert or delete a string value to or from the environment space. The new string value(s) may be used by DOS and application programs to help perform various functions. The syntax for the SET command is shown in Figure 8.

```
SET                         (Display environment values)
SET name=string             (Define string value)
SET name=                   (Delete string value)
```

Figure 8
SET Command Syntax

If you enter the SET command by itself, it will display the current values in the environment space. Assuming that neither a PATH nor a PROMPT command has been executed, you will see a display similar to:

```
PATH=
COMSPEC=A:\COMMAND.COM
```

The first line showing only "PATH=" means that there is a "null path", and if you enter the PATH command by itself, it will display the "No path" message. The second line of the SET display with the COMSPEC= parameter tells DOS exactly where to find the Command Interpreter COMMAND.COM. This example assumes a floppy disk system so that DOS will look in the root directory on drive A to find the file name COMMAND.COM. If DOS cannot find it as specified in the COMSPEC= line, you will see a message similar to "Insert disk with \COMMAND.COM in drive A and strike any key when ready" indicating that DOS tried to reload the transient portion of COMMAND.COM and could not find it in the root directory on drive A.

The second command line is used to set a defined *name* equal to a specified text *string* value. This command form can also be used to replace an existing text string with a new string value. For example, you might want a programming compiler to use a virtual disk with drive letter D for temporary storage of files, and you might define that with the command:

```
C:\ ==>SET TMP=D:
```

Note that any software using this feature must be especially designed and programmed to access the environment space to obtain information. In this example, the compiler would have to be specifically programmed to look for the "TMP" name with the associated drive letter.

The third command line is used to delete the *string* value for a specific *name* from the environment. In the previous example, you could delete the "TMP" value with:

```
C:\ ==>SET TMP=
```

If you need to use a SET command, the documentation for that application should tell you what to do and the syntax required to do it. You may want to include all required SET commands in the AUTOEXEC.BAT file or you may want to set up a special batch file to start that particular application.

The SET command can be entered in either uppercase or lowercase letters, but there is one peculiarity you should know about. The *name* value is converted to uppercase because it is stored in the environment that way, but the *string* value is NOT converted to uppercase -- it is stored EXACTLY as entered. Whether or not the lowercase entry causes a problem is dependent on your specific application for the *string*, but you should remember that the string value is stored exactly as you type it.

If you need to use the SET command, you may see the "Out of environment space" error message, particularly if you have already used the PATH and/or PROMPT commands. Let's find out what that really means and then see how to fix it.

The Environment Space Limitation

To understand the space limitation in the environment, it is necessary to briefly consider how COMMAND.COM is designed. The Command Interpreter basically has three different parts: the initialization portion, the permanent portion, and the transient portion. The initialization portion is only used during the boot process and is responsible for such things as the AUTOEXEC.BAT file execution. The permanent portion is permanently resident in your computer's system memory in a "low address" that is close to the beginning of RAM or 0 K. As a matter of fact, this part of low memory contains the BIOS, System Kernel, and the resident part of COMMAND.COM. The permanently resident part of COMMAND.COM is memory-resident in much the same way as the TSR programs that have been previously mentioned in these articles.

The last part of COMMAND.COM, the transient portion, is loaded into "high memory", and if you have a total of 640K RAM, you would find that this part of the

Command Interpreter begins about 635K or so depending on your DOS version. This portion is transient — not permanent — because an application program can actually use this memory if needed and destroy the transient portion of the Command Interpreter. For that reason, DOS always attempts to reload the transient part of COMMAND.COM, and the whole purpose of the COMSPEC= line that you saw with the SET command tells DOS where to find COMMAND.COM so that it can be reloaded.

The environment space is limited because it is actually a "piece" of the memory used by the transient portion of the Command Interpreter. Unfortunately, it is not a trivial matter to simply expand that particular piece of memory because DOS has no way of knowing if you have loaded another TSR program just below the transient portion of the Command Interpreter. Each TSR program normally determines the top of free memory and uses this as the upper boundary for its program code. The first TSR program loaded is normally just below the transient COMMAND.COM. The second TSR program is normally loaded just below the first, and so on. Note that *you* determine the order of the loading by the order in which you enter the commands. That can cause some interesting problems in a few TSR programs.

A few of these memory-resident utilities do not have a fixed size requirement for memory space. That is, the lower boundary varies depending on the function(s) being used. You can easily recognize these programs because the documentation clearly states that these programs must be loaded last. This is not the only reason to load a program last, but it is one of them.

One of the reasons for introducing this technical discussion of how DOS uses the environment space is to help you understand why there is a size limitation in the first place. Now let's see how to fix it.

The COMMAND Command

Many users are not aware that the COMMAND command can be executed just like any other DOS command. The COMMAND command is obviously external and it loads a secondary copy of the Command Interpreter. While that may not seem to be especially important to you at this point, I will show you how to easily solve a very common batch file problem that many users have when they try to develop complex batch files. The usage requires that you have at least DOS version 3.10 or later so that you can use the syntax shown in Figure 9.

```
COMMAND/C [d:][path]your-command    (Execute and EXIT)
COMMAND/E:nn[/P]                    (Environment space)
```

Figure 9
COMMAND Command Syntax

In the first command line, the /C (Command) switch loads a secondary copy of the Command Interpreter, executes *your-command*, and exits back to the previous program which is usually the transient portion of COMMAND.COM (i.e. the DOS prompt). Note especially that this command form with the /C switch returns to the "program" that was running at the time this command line was executed, and this "program" may be a batch file in addition to the usual COM and EXE programs. You will see why this is an important feature for a batch file later in this article.

The second command form is used to load a secondary copy of the Command Interpreter and expand the environment space with the /E (Environment) switch. In many releases of DOS version 3.10, the number of bytes for the environment *nn* is specified in memory paragraphs, and the default is 10. For both memory specifications and programming, a PARAGRAPH is defined as 16 bytes. Since the *nn* default is 10 for these DOS versions, the environment space default is 160 bytes. The valid range for *nn* is 10-62 paragraphs which gives you an environment space of 160-992 bytes.

For most DOS versions 3.20 and later, the *nn* value is simply specified in decimal bytes. The usual range is 160-32768 bytes although the default is 128 bytes in some of these versions. Check the COMMAND and SET commands in the DOS manuals for your specific version since most of the details can be found under one of those commands. If your manual does not specifically mention the /E switch (e.g. the Z-100 manual for version 3.10 does not), it is probably not implemented and will not work. In some Zenith MS-DOS manuals, you will find the default listed under the COMMAND command, but many of the manuals incorrectly indicate that the default is 128 bytes even though it is actually 160 bytes in most current releases. If you attempt to enter an invalid value for the environment space value *nn*, you will see an "Invalid environment size specified" error message displayed.

The optional /P (Permanent) switch is used to load the secondary copy of COMMAND.COM permanently in memory so that it cannot be "removed" without rebooting the system. In general, you should not need to use this option, but J have included it because it is a standard feature of current DOS versions. One reason for not using the /P is because there is no easy way to recover the memory used until the system is rebooted.

Be careful when you use the /E switch because it creates a copy of part of COMMAND.COM in high memory. In other words, this form of the COMMAND command really is a memory-resident (or TSR) program. The actual amount of memory used depends on your DOS version (as usual) as well as the number of bytes specified for the environment space. Be extremely careful when using COMMAND with the /E switch if you have other TSR programs because COMMAND is also memory-resident.

Also note that COMMAND with the /E switch ***stays*** resident in memory until you change it, even if you did not use the /P (Permanent) switch. If you used the /P switch, the secondary copy of COMMAND.COM will stay in memory until you reboot the system. But if you did not use the /P switch, you can release that memory used by using the EXIT command if you are careful.

The EXIT Command

EXIT is an internal DOS command that terminates execution of the secondary copy of COMMAND.COM, releases the memory used, and returns system control to the previously running program which is normally the primary COMMAND.COM located in high memory. The syntax for this command is shown in Figure 10.

```
EXIT
```

Figure 10
EXIT Command Syntax

EXIT *ONLY* terminates the secondary copy of the Command Interpreter when you have used the /E switch to change the size of the environment space, and it only works when the secondary copy has not been permanently loaded into memory with the optional /P switch. The EXIT command has no meaning when used after the /C (Command) switch because that switch causes the Command Interpreter to automatically exit after the execution of *your-command* shown in Figure 9.

The EXIT command has absolutely no effect on the primary (i.e. transient) copy of COMMAND.COM that is located in high memory. It doesn't cause any problems if you try it, but it just doesn't do anything.

It is important to be very careful when you use the EXIT command because it can cause a problem that is not mentioned in any DOS manual I have ever seen. Remember that EXIT terminates execution of the secondary copy of COMMAND.COM, releases the memory used, and returns system control to the previously running program which can be either COMMAND.COM or an application program, but here's what can happen if you forget.

If you decide to expand the environment space with the /E switch (without entering the /P switch also) and have loaded other TSR programs after that, do NOT use the EXIT command. If you do, there will be a "hole" in the memory space, and all kinds of unpredictable and undesirable things can happen. In other words, do not use the EXIT command unless COMMAND was the last TSR program you loaded into memory.

At this point, you know how to fix the "Out of environment space" problem with the COMMAND command, and how and when to use the EXIT command. Now we will take a look at a much more common application of the COMMAND command that will help you develop more useful and complex batch files.

Batch Files and the COMMAND Command

There are certain occasions when it would be useful to have one batch file be able to execute another batch file, and then return to the next line in the original batch file. This technique is usually called CHAINING because of its similarity to the CHAIN command in BASIC, and it is normally used when you want to have a general "menu" batch file "call" a startup batch file for a specific application. Although this seems like it would be an obvious application, you will not be able to do this unless you know the trick. And the DOS manual does not really offer much of a clue about this either.

Let's say that we wanted to create a simple batch file called START.BAT that might contain lines shown in Figure 11.

This simple batch file is intended to call either WORDSTAR.BAT or SUPERCAL.BAT depending on whether the command line contains W or S. This makes it easy to create and use specific batch files for each one of your programs. The batch file for a specific application (e.g. WORDSTAR.BAT or SUPERCAL.BAT) contains commands to change to an appropriate subdirectory, install a keyboard macro utility, and make an automatic backup copy of the document to a floppy disk.

If you try to use a batch file like the above, it will stop executing once the selected "IF %1==" line is run, and the following ECHO lines will never be executed. This example is a simple one, but it illustrates that, after the primary Command Interpreter "calls" a second batch file, it "loses its place" in the first batch file, and the remaining lines will not be executed. This happens with batch files because they are not loaded into memory like a COM or EXE program is. Let's fix that problem with one simple change.

All you have to do is add the COMMAND/C in the appropriate place where a second batch file is called as shown in Figure 12.

This addition of the "COMMAND/C" in each line loads a secondary copy of the Command Interpreter so that it remembers the place where it left the original batch file. All you have to do is add the COMMAND/C before each call for a batch file, and you can continue to execute multiple levels of batch files if you are so inclined. Now the batch file will execute the last two ECHO commands or any others that you care to add.

The batch files presented here were for illustration only, and they are not good examples of how to write a batch file. For example, each "IF" statement should have been duplicated with a lower case "w" and "s" so that a user would not have to be sure to enter those values in capital letters. The whole idea was to present the technique and syntax requirement to successfully execute multiple batch files.

Powering Down

As we have taken a brief look at various PC compatible application programs and DOS commands in this series, I have not attempted to discuss each and every available DOS command nor have I included all possible options in the commands which were covered. The primary purpose of this series was to tell you about the various commands that you must know (e.g. FORMAT, COPY, etc.) and familiarize you with various commands that can help you solve a common problem with a specific focus on Zenith MS-DOS. There are also some differences in commands, particularly new optional switches in the current releases, in the various DOS versions which is the reason that many of these discussions only mentioned the common and most useful options.

There is one other very important reason for limiting these command discussions to common options and functions for each command. As we move toward version 4 of DOS, many of the details of these command options may change, but the information included in this series will probably be valid for DOS version 4 and beyond. As always, be sure to take a look at your DOS manual for additional information on these and other commands for your specific version.

Next Time

Computer maintenance is one topic that seems to be consistently overlooked in most articles and books. Some maintenance, such as keeping the CRT clean, is obvious, but there are some things you should know about even that simple task. In the next article, you will learn about the various kits you can buy to help you maintain your system, and you will see how to troubleshoot and fix some common problems.

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

MS-DOS Version 3.3+ (OS-51-3) $149.00
Heath/Zenith Computer Centers
Heath Company
Hilltop Road
St. Joseph, MI 49085

*

Continued from Page 4

finds in unused areas of the disk so they will not be used as you add more data. This is a new version of DTEST that supports partitions larger than 32 megabytes.

SCLK.COM — This is a new version of the HUG screen clock program that works great with laptop computers (works great with desktops too). It maintains a digital clock display in the upper right corner of your screen while you run other programs. This version of the screen clock will not interfere with your cursor setting (whether done by the CURSOR program or another way), or any other program on this disk.

Table C Rating: 10

*

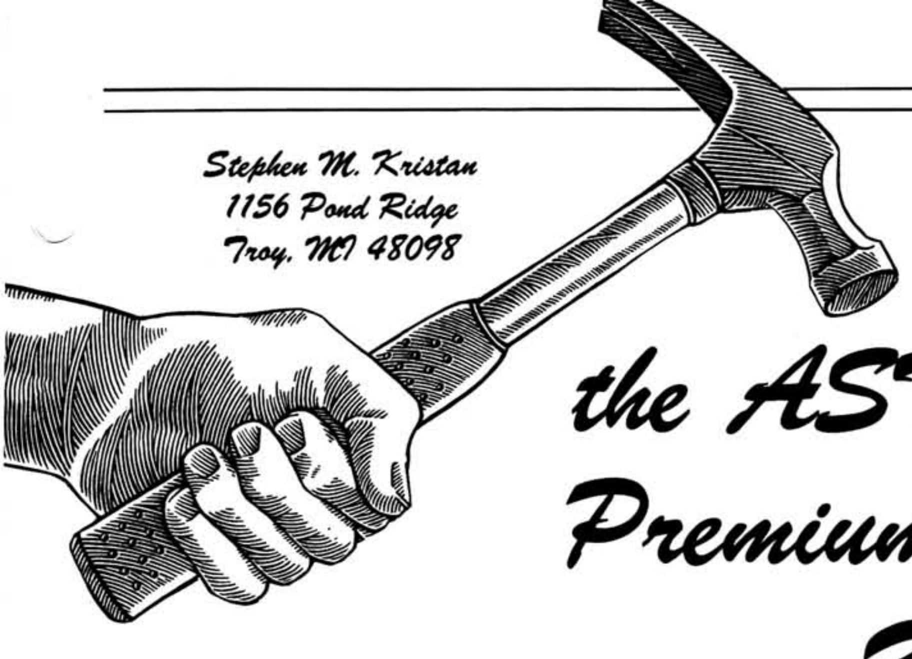
```
ECHO OFF
REM This batch file will start a word processor or
REM spreadsheet depending on the command line options.
ECHO For word processing: START W filename.typ
ECHO For spreadsheet :    START S filename.typ
IF %1==W C:\WS\WORDSTAR
IF %1==S C:\SC\SUPERCAL
ECHO Word Processing or Spreadsheet completed
ECHO ON
```

Figure 11
Batch File without COMMAND/C

```
ECHO OFF
REM This batch file will start a word processor or
REM spreadsheet depending on the command line options.
ECHO For word processing: START W filename.typ
ECHO For spreadsheet :    START S filename.typ
IF %1==W COMMAND/C C:\WS\WORDSTAR
IF %1==S COMMAND/C C:\SC\SUPERCAL
ECHO Word Processing or Spreadsheet completed
ECHO ON
```

Figure 12
Batch File with COMMAND/C

Stephen M. Kristan
1156 Pond Ridge
Troy, MI 48098



Installing the AST Advantage Premium Board in a Z-200

Computer memory is like money. You can never have too much. Whether it's needed to handle more sophisticated software or simply to squeeze more performance out of an existing machine by use of a ram disk, one of the first enhancements you'll probably buy for your computer system is a memory upgrade. In designing the new Z-248/12 and Z-286LP Zenith made adding memory to these machines very easy. It's simply a matter of adding memory chips to the motherboard. (Up to 6MB will fit on the motherboard.)

However, it's not as easy to add memory to the older Z-241, 248, and 286. Instead of a motherboard, these machines have a backplane. The CPU and system memory reside on a board which mounts onto the backplane. Since only 512K of memory fit on this board, additional memory must go on a separate memory board. This article tells you how to install an AST Advantage Premium memory board in one of the older machines.

For the Z-241, Z-248 and Z-286 machines Zenith offers an add-on memory board that brings base memory up to 640K. After that, additional Zenith memory boards can be used as extended memory. Here's the critical distinction between the Zenith memory board and various third party memory boards. While Zenith boards only give you extended memory, which Zenith refers to in SETUP as "expansion memory", several third party manufacturers offer memory boards that can be configured as either extended or expanded memory.

Before we go any further, let's take a look at the types of memory available on an MS-DOS-based machine. When the 80286 microprocessor first came out, it had three areas of memory. From 0K to 640K, called conventional memory, was set up to be user memory in which DOS and the application program resided. Memory above 1MB was called extended memory. Although application software could not use this extended memory, it could be used for RAM disks and print spoolers. The area between 640K and 1MB was not accessible at all. It was used for system functions, such as video RAM and system ROM.

Over time, memory requirements increased. Application programs became more sophisticated and required more memory. Lotus, Intel, and Microsoft (LIM) came out with the Expanded Memory Specification 3.2 (EMS LIM 3.2) that allowed the computer to use more than 640K. Technically, MS-DOS still had the limitation of not being able to use more than 640K at one time. EMS allowed you to use memory between 640K and 1MB to store data and swap or page data in and out of the original 640K. Software designed to support LIM 3.2 uses this expanded memory area to store data, but not the actual program code. The end result of EMS was that software was no longer limited to 640K, but rather under LIM 3.2 you could access up to 8MB of EMS memory.

Following LIM 3.2, AST, Quadram, and Ashton-Tate developed Enhanced Expanded Memory Specification (EEMS) which added bigger pages to swap and the ability to swap part of the program, not just the data. This additional functionality requires additional components on the memory board itself. Since EEMS was an enhancement to EMS, EEMS memory boards still support EMS. The reverse is not true. An EMS board will not support EEMS.

In 1987, LIM 3.2 incorporated many of the functionalities of EEMS and became known as LIM 4.0. The biggest advantage to LIM 4.0 is the ability to run program code in expanded memory. Total amount of expanded memory that could be accessed went up to 16MB. Like EEMS, LIM 4.0 requires additional hardware components on the memory board.

Some older LIM 3.2 boards could be made LIM 4.0 compatible by adding new LIM 4.0 software drivers, but the performance speed of these older boards would be less than that of the boards originally designed for LIM 4.0. These LIM 4.0 software drivers had the effect of fooling the application software into thinking that the memory board was LIM 4.0, when in fact it was not.

EEMS boards are different. Since the LIM 4.0 specification was closely related to EEMS, boards that were originally designed for EEMS are hardware compatible with LIM 4.0. By simply adding LIM 4.0 software drivers, existing EEMS boards can become fully hardware compatible with LIM 4.0.

LIM 4.0 is now the emerging established standard for expanded memory and is fully supported by Zenith in the current Z-286LP, Z-248/12 and Z-386 machines. However, the Zenith memory boards for the older Z-286 and Z-248 do not support EMS (either LIM 3.2 or 4.0) nor EEMS. To run expanded memory or EEMS on these machines, you need to add a third party memory board.

One of the most popular third party boards to use in the Z-248 and Z-286 is the AST Advantage Premium. This board comes standard with 512K of memory. A portion of this memory can be used to backfill the original 512K to raise conventional memory to 640K. The remainder of the AST memory can be configured as EMS, EEMS or extended memory. In addition, the board also comes with a serial port, parallel port, and optional game port.

The AST Advantage Premium board's major "advantage" is that it was designed to support EEMS. Adding the requisite LIM 4.0 software driver enables it to become hardware compatible with LIM 4.0. Remember, this is possible because it was designed to support EEMS. Another advantage is what AST calls "split memory addressing". Most memory boards with 256K memory chips actually view memory in 256K blocks. This means while you only need 128K to backfill your 512K machine to 640K, the memory board will see the 256K blocks, use 128K of that to get 640K and not use or, in essence, waste the other 128K. Split memory addressing allows the AST board to take that unused 128K and fully utilize it as expanded or extended memory.

The AST Advantage Premium board comes in a variety of sizes. The base unit comes with 512K. This can be expanded to 1MB by adding memory chips to the main memory board. An optional half card daughter board plugs into the main board and adds another 1MB for a total memory capacity of 2MB. The daughter board overlaps half of the adjoining slot, but this might not be a problem. In my machine, the slot next to the Advantage Premium is taken up by my Microsoft bus mouse card, which takes only a half slot. The end result is that I get a memory board with the capacity of 2MB, second serial and parallel ports and a bus mouse card and only give up 2 slots.

For my system, I purchased the AST Advantage Premium board with 512K of memory. I then added another 512K of memory chips to get 1MB on the memory board. This 1MB would be used to backfill my system memory to 640K with the remaining memory being used as LIM 4.0 memory.

Installing the Board

The first step in the installation process is to set the memory board dip switches. The three sets of switches are located in the middle of the top row on the board. Switch 1 is used to allocate conventional and extended/expanded memory. Switch 2 is used to tell the AST board how much memory is already installed in your machine. Switch 3 is used to configure the serial and parallel ports.

Setting the switches is uncomplicated. Leave switch 1 at its default settings. SW1-1 through SW1-4 come from AST set to 128K. This means that 128K will be used to backfill the system memory to 640K. Switches SW1-5 through SW1-8 set up the base I/O address at 0218h. Switch SW1-9 enables Dual Page mode and switch SW1-10 disables the second Advantage Premium serial port.

The AST documentation for Switch 2 is a little bit confusing. The manual says that this switch tells the AST board how much combined conventional and extended memory is already installed in your machine. At first, this sounded silly since, if I already had extended memory, why would I need the AST board in the first place. Then I remembered that a machine can have multiple Advantage Premium boards installed. Since this is to be the first AST board installed in the machine, the total conventional and extended memory is only 512K. Keep the default setting of 512K.

Switch 3 is used to set up the serial and parallel ports. Since the Zenith machine already has one serial port, the AST serial port should be configured as COM2. This means setting SW3-1 to OFF and SW3-2 to ON. Next, since Zenith also has one parallel port, the AST board parallel port should be configured as LPT2. That is done by setting SW3-4 to OFF and SW3-5 to ON. Finally, the interrupt for the AST serial port needs to be set. In my machine, the bus mouse board is set for IRQ5 so I set the AST serial port to IRQ3 by setting SW3-7 to OFF and SW3-8 to ON.

The AST board also has several jumpers used to enable or disable the serial and parallel ports. The default for all these is enabled. Keep them at the default settings.

Once all the dip switches are set, you are ready to install the board into the computer. Remember, the only switch I changed from the factory default settings was switch 3. Now disconnect the power cord from the computer and remove the cover. As a safety precaution, it's a good idea to always disconnect the power cord anytime you are adding or removing a board in your PC. Check your Zenith owner's manual to find out which screws need to come out to remove the cover.

Next, select an empty slot to put the AST board into. It must go into a 16 bit slot, not an 8 bit slot. The Zenith manual refers to the 16 bit slot as a PCAT slot. Unscrew the mounting bracket face plate from the empty slot and remove it. Carefully slide the AST board into the slot. You will need to push firmly down on the board to insure that it goes in all the way. Be careful to push straight down so you don't break the board. Once in, secure the board by screwing down the mounting bracket.

Next, slide the cover back on the machine, but don't put the screws in yet. Should you need to make any dip switch changes, you'll only have to unscrew them once again. Plug in the power cord, turn the machine on and go into the setup ROM monitor program.

When the computer boots, it checks the setup monitor program located in ROM to see how much memory it should have. It then physically checks the machine to see how much memory is actually installed. These two numbers need to match. If they don't, you'll get an error message saying that setup and actual memory amounts don't match. Bring up the monitor program by pressing [Ctrl] [Alt] [Ins]. At the prompt, type in SETUP. Change the Base Memory Size to 640K, but leave Expansion Memory to 0K. This can be confusing. The Zenith machine doesn't know anything about EMS, only extended memory. When setup asks for the amount of expansion memory, it's referring to extended memory. The AST board was set up for expanded memory, not extended memory so leave the setup expansion memory set at 0K.

The AST board allocates memory as expanded or extended by using device drivers in the config.sys file. To set up this memory allocation, run the install program off of the AST diskette included with the AST board. When you run the install program and select expanded memory, the install program will copy a program called REMM.SYS to your config.sys file.

According to the AST manual, the remm.sys file is the software driver that creates the pointers, loads the registers, and maps the PC windows to Rampage expanded memory.

To run the install program, insert the AST diskette that comes with the board into the A drive and type INSTALL <ENTER>. Next, you'll be prompted for your boot drive. Type in C <ENTER>. Next, you'll be asked questions regarding the type of video board you are using. On my Z-286, I have the Z-449 video board which is considered an EGA board so I selected Y when asked if I was using an IBM Enhanced Graphics Adapter or compatible. It's important to answer this question correctly so that the AST board doesn't try to access memory used by the video board.

Once the install questions are answered, the main menu appears indicating how the board was configured. Under the Expanded Memory Configuration section you should see an I/O port address for Board #1. Boards 2, 3, 4 will not have addresses since you only have one AST board installed. Conventional memory should indicate 640K. Expanded memory should list all your add-on memory, less the 128K needed to backfill. In this particular example, since we only wanted expanded memory, the total extended memory should say 0K.

The AST board also provides capability for RAM disks and a print spooler. These are referred to as FASTdisk, SuperDrive, and SuperSpool. These options are also configured by running the AST install program.

Once the install program is com-

Continued on Page 26

Hard Disks and Bernoulli Drives for the Z-100

Dixie D. Green
600 Las Posas Street
Ridgecrest, CA 93555

Introduction

This article was written to provide my fellow Z-100 enthusiasts with some information on the addition of two different mass storage devices to the Z-100. The first mass storage unit is, of course, the hard disk and the second is the Bernoulli Box. This particular hard disk addition applies only to the Z-100 All-in-one, but the Bernoulli Box addition is applicable to both Z-100 models. The hard disk and Bernoulli Box additions were done on separate Z-100 computers.

My reason for covering the hard disk is that, to my knowledge, this particular hardware has not been covered previously nor has this implementation. I am not aware that any article has covered the interfacing of the Bernoulli Box to the Z-100.

Although I have tried to be accurate in my statements, please verify all steps before proceeding. The responsibility for any implementation is yours.

Interfacing the Hard Disk

I had never intended to add a hard disk to my Z-100 All-in-one; I had found that the two eight inch drives, in addition to the two floppy drives internal to the Z-100, offered more than adequate storage to meet my needs. However, the addition of a Gemini board left me wanting for storage space when operating in the PC mode, since the eight inch drives are not supported by the Gemini. However, I did not wish to sacrifice one of my internal disk drives or to add an external piece of hardware in order to add the hard disk.

My Z-120 is configured with the HA-108 upgrade to 768K of memory and 8 MHz operation. It has half-height floppies and the "new" motherboard. This portion of the article is not for those with full-height floppies, as will soon become apparent.

My choice of a hard disk controller board was the Konan DGC-2000. The complete package including controller board, hard disk drive, cables and supporting software is available from Lindley Systems and requires version 2.0 or later of MSDOS. Lindley Systems will also sell the board and software as a package, for those of you who might save money by doing a little comparison shopping for your hard disk drive. I purchased my Seagate ST225 hard disk from Scottie Systems and my cables from Altex Electronics.

If you have the Gemini board, it is necessary to upgrade your EPROMS (erasable programmable read-only memory) to version 1.5 to support the device drivers supplied by Lindley. You may purchase these from Lindley for $30. The software I received contained device drivers for the Z-100, the Gemini board and the UCI board. I requested and received the source code for the device drivers. It should be obvious that since device drivers are required, this software will not support CP/M.

Installation fo the Hard Disk

Refer to your hardware documentation manual for disassembly instructions. The information is listed in the General Information section. You must first remove the top and then the Display and Disk Drive Assembly. After I removed the Display and Disk Drive Assembly, I placed it face down on a piece of foam. This protects the face of the Display and Disk Drive Assembly.

To mount the hard disk, you must first remove the metal housing from the remainder of the Display and Disk Drive Assembly. Remove the ten screws holding the metal housing, four in the bottom, two in each side and two at the top. The entire disk drive assembly may then be removed from the rest of the assembly. On the sides of the metal housing for the disk drives, near the bottom, you will find four holes identical to the ones utilized in mounting your disk drives. This is the position for the hard disk. Notice that there is a cooling slot in the front plastic panel of the Display and Disk Drive Assembly. Before mounting the hard disk in the metal housing, I performed two modifications.

First, I removed the faceplate from the front of the hard disk so that air flow around the drive would be less restricted. The second modification was performed so that I could observe the LED (light emitting diode) activity light on the hard disk. You might wish to skip this modification. Note that the hard disk is to be mounted with the circuit board downward. I removed the plastic front from the metal housing, and directly in front of where the LED on the hard disk would be located, I cut a hole in the plastic. To make this modification a little more appealing, I filed the overhang above the cooling slot and around the hole until it was flush with the surface bearing the Zenith Data Systems logo. I then glued a rectangular piece of translucent plastic, obtained from a push-button switch, over the hole and extended across the cooling slot. There are numerous alternative methods of accomplishing the same result, including mounting the LED on the front panel or cutting a hole for the plastic faceplate of the Seagate. Next replace the front cover on the metal housing.

If you have the Gemini board, you should now replace the EPROMS. Refer to the instructions that will accompany your new EPROMS.

Before mounting the hard disk in the metal housing, check to insure any necessary jumpers on the hard disk are installed. The Seagate ST225 requires a jumper between pins 15 and 16 on J7 to configure it as drive 1. Also note the position of pin 1 on J1 and on J2. When the hard disk is installed with the circuit board downward, pin 1 will be the left-most pin on the side toward the top of the computer. J1 will be to the left and J2 to the right, viewed from the top of the computer. Next, mount the hard disk in the metal housing, circuit board downward.

There are three cables to be attached to the Seagate ST225. There is one four conductor cable for power, one twenty conductor ribbon cable and one thirty-four conductor ribbon cable. The connection of the ribbon cables between the hard disk drive and the Konan board is well covered in the instructions that accompany the package and I will not cover it here, other than to give the information necessary for ordering the cables. The ribbon cable assemblies are available from Altex, as part numbers 20S-20E and 34S-34E. You need to specify the length of these cables when ordering; mine are one foot long.

However, since the connection of power to the drive differs from that anticipated by Lindley Systems, I will cover that topic. There are two sources of power available for the disk drives and the hard disk. These are P4 and P5, normally connected to drives A and B. To furnish power for the hard disk, one of these two is used and the remaining one is used to furnish power to both drives A and B. This requires an adapter cable assembly. This adapter cable is part number YAD-4, available from Altex Electronics. I used P4 for the hard disk and P5 for the floppies.

Begin reassembly of your computer by placing the Display and Disk Drive Assembly on the base of the Z-100. Make the power connections by plugging P4 into the hard disk and the Y adapter cable assembly into P5 and drives A and B. Then, reconnect the ribbon cables to the disk drives.

Before installing the Konan board, you need to check the position of two jumpers on the Konan board. The procedure for doing this is outlined in the accompanying literature and will not be repeated here. Finish reassembly of your computer and continue to the software installation.

Initializing the Hard Disk and Installing Drivers

The software furnished by Lindley Systems included the device drivers mentioned previously and four hard disk utilities. These are DGPREP.EXE, DGPART .EXE, DGPARK.EXE and DGFIX.EXE. **All these utilities must be run under the Z-100 mode.**

To install the device drivers, you must copy the driver DGZ.DVD to your boot disk for the Z-100 and add the line

```
DEVICE=DGZ.DVD
```

to your config.sys file on the boot disk. Likewise, you must copy the device driver DGG.DVD to your boot disk for the Gemini mode and add the line

```
DEVICE=DGG.DVD
```

to your config.sys file on your Gemini boot disk.

DGPREP.EXE is the utility used to initialize your disk. It formats the disk and creates the MSDOS file structure. It is interactive with the user; it presents several questions and prompts for the answer. In my case, Lindley Systems had attached a gummed label to the bottom of the page with all the setup information for my Seagate ST225. The version of DGPREP .EXE that I received was version 1.03 and required approximately seven minutes to complete. The manual states that at completion of the formatting, you will be asked if you would like to do a media check. My version did not do this and on checking with Lindley, I was told that it was now accomplished automatically during the formatting process. The disk is completely formatted by this utility. The Heath/Zenith programs FORMAT, PREP and PART are not necessary and will not work with the device driver. The H/Z program DETECT is also not necessary since the disk controller board automatically handles any defective blocks found on the hard disk. However, the H/Z programs BACKUP and RESTORE will work with this driver.

DGPART.EXE is the utility used to created the partitions on your hard disk. The strong point of this software lies here in the fact that you can create partitions that are accessible to the Z-100 mode, the Gemini mode or both. The performance of the DGPART utility differs slightly from that in the documentation, but I attribute that to the fact that the documentation is for version 2.02 and I received version 2.05. There were slight differences in the visual menu and my drive was partitioned to slightly less length and byte capacity.

DGPARK.EXE is the utility that parks the disk drive heads at a track inside the last available track used for data. You should run this utility before shipping or any time you suspect your computer will be subjected to rough handling. The heads are automatically "unparked" during normal power-on sequence.

DGFIX.EXE is a sector editing utility and its operation is not covered in the documentation furnished by Lindley. Whenever I attempted to run it on my computer, it would always result in the message that no drives were on line.

Problem Areas

I encountered two problems in addition to DGFIX.EXE. The first problem appeared to be noise related and the second related to the thermal environment within the Z-120. Both these problems only appeared when operating in the Gemini mode. Therefore, I ruled out the hard disk drive and the Konan controller as sources of my problems.

The first problem appeared in the form of a faint flicker on the hard disk activity LED whenever the computer was not accessing a disk drive. The second problem occurred after the Z-100 had been operating and had "warmed up". When rebooting, the device driver would refuse to recognize the hard disk and would return the error message "No drives on line." In discussions with people at Lindley Systems, I was assured that it could not be the EPROMS and that other people had experienced similar problems that were resolved when their Gemini boards were returned for repair or exchange. Before doing so, I spoke to the people at Gemini Technology and they suggested that I enable two wait states instead of the one wait state required by my HA-108 upgrade. When that failed, I returned my Gemini board for repair or replacement.

When my Gemini board was replaced, I expected both of the problems to have been solved; yet the problems were still there. When I installed my upgraded Gemini EPROMS in a second Z-100 with a Gemini board, neither problem appeared. When I installed my Gemini board with upgraded EPROMS in the second Z-100, the problems were still absent. Thus, the only remaining possible source of the problems was my 8088-2 microprocessor, the one furnished in my HA-108 upgrade. When the 8088-2 was replaced, the problem with the flickering LED disappeared. The thermal problem was a little more difficult to isolate. The upgrade EPROMS (U27) were found to be marginal at 8 MHz after warming up, but they worked fine in the second Z-100 which was operating at 5 MHz. After I copied the contents of the Gemini upgrade EPROMS into a higher quality, faster EPROM, the device driver never failed to recognize the hard disk.

Interfacing the Bernoulli Box

The Bernoulli Box can come with either of two interfaces. One is with an IBM-PC interface card and the other is as a standard SCSI (Small Computer System Interface) used by Apple. The Bernoulli Box that is interfaced to the Z-100 has the SCSI interface.

The controller used for the Bernoulli Box is the CDR317-IIB from C.D.R. Systems, Inc. The Bernoulli Box is the Iomega 10+10, an eight inch dual 10 megabyte drive.

Two comments on configuration; if you have a Gemini board, you must have Version 2.0 of the Gemini EPROMS and the EPROM on the CDR317-IIB must be for the Bernoulli Box. The EPROMS for the Gemini are available from C.D.R. Systems, Inc. for $35 and you should order the CDR317-IIB configured for your Bernoulli to insure you have the correct EPROM on the controller board.

Hardware Installation

If you have the Gemini board, your first step should be to install the newer version EPROMS. Refer to the instructions given earlier in the article for installation of Gemini EPROMS.

Installation of the remaining hardware is a simple matter. First, you must set the SCSI device number to zero on the Bernoulli Box. This is accomplished by finding a rotary switch on the rear of the Bernoulli. This switch has a slot for a screwdriver and a small indentation for an indicator. The switch is turned until this indicator points to zero. There are two SCSI connectors on the back of the Bernoulli, one for each drive. They are connected together inside the box, so connection between the Bernoulli and the Z-100 may utilize either SCSI connector. A cable terminator is then placed on the other SCSI connector. This Bernoulli was intended to be used with a Macintosh and the cable terminator was included. The documentation indicated that the terminator may be obtained from Apple dealers.

The cable that is bundled with the Bernoulli was not used since its computer connector was incompatible. You may construct this cable by using a 50-pin male Centronics (57 series) type connector for mating to the SCSI connector and a 50-pin IDS socket for mating to the CDR317-IIB. I suggest using ribbon cable in the manufacture of the connecting cable. For this choice, the connectors are available from Altex Electronics as part numbers ID57-50P and IDS50. The cable is manufactured with pin 1 on one connector connected to pin 1 on the other connector.

The cable may be pushed through one of the larger slots in the rear panel of the Z-100 and connected to the CDR317-IIB. The documentation accompanying the CDR317-IIB contains all the information required to configure the controller for use in the Z-100. You must install the jumper at JJ2 to the A position to enable formatting of the Bernoulli cartridges. After all cartridges have been initialized, you may wish to place this jumper to the B position to enable format protection.

After configuring the controller board, make the cable connection with the Bernoulli and install the controller board in a convenient S-100 slot. You are now ready to initialize the Bernoulli cartridges.

Software

The software included with the CDR317-IIB controller included copies of the Zenith Winchester Utilities PREP .COM, PART.COM, DETECT.COM, VERIFY.COM, and SHIP.COM. These utilities come with the notice that they are "on loan" and please purchase your own copies when they become available. Normally, Zenith only sells the Winchester utilities with the Z-100 hard disk upgrade or with a hard disk equipped Z-100. Zenith has posted a notice on their bulletin board that the Winchester utilities are now available for purchase. You should send a written request to:

Zenith Data Systems
Attn: Jane Eaton
Hilltop Road
St. Joseph, MI 49085

Your written request should include name, address, phone number, serial number of your Z-100 Demo disk (if available), a $25 check made payable to Zenith Data Systems and a statement explaining why you are unable to return your original copy of the Winchester Utilities Disk.

Also included were the utilities FASTPREP.COM and BERNPREP.COM. Both FASTPREP.COM and BERNPREP.COM are variations of PREP.COM. The number of passes made in initializing the cartridge is reduced from seven for PREP to three for FASTPREP and to one for BERNPREP.

Initializing the Bernoulli Cartridge

Insert a cartridge in the Bernoulli Drive. If the Bernoulli is a dual drive unit, insert a cartridge into each drive. Consult your manual for directions on loading the cartridges. Power up the Bernoulli and reboot your computer. All software is (first) executed with the Z-100 in the Z-100 mode. Therefore, if you have either a Gemini or UCI board installed in your Z-100, be sure you choose the Z-100 mode of operation on bootup. Later, similar utilities may be executed using H/Z-150 PCDOS and CP/M-85.

Although information on the utilities used to initialize your hard disk may be found in your Z-100 manuals, the Gemini User's Manual and the HELP screen available on power up of your computer, I had difficulty in successfully completing the initialization of the Bernoulli cartridges by following these instructions. Therefore, the instructions from these sources have been modified, based upon my experiences and experiments.

With the Winchester Utility disk on your default drive, execute the FASTPREP (or BERNPREP) utility by typing

```
FASTPREP/Q  (or BERNPREP/Q if using
                            BERNPREP)
```

You will be prompted to answer some questions concerning your drive. One of the questions you must answer is the drive number. Respond with the number zero and note which Bernoulli drive becomes active as FASTPREP is executing. This is drive 0 and the other (if you have two) is drive 1. It will be necessary to know this when you assign the drives later. The remaining prompts and your responses are given below. The responses for a Bernoulli 20+20 are given. Where the response for a Bernoulli 10+10 is different, it will be given in parenthesis.

NUMBER OF HEADS:	4
NUMBER OF CYLINDERS IN HEX:	243 (113)
REDUCED WRITE CURRENT CYLINDER IN HEX:	0
PRECOMP CYLINDER IN HEX:	0
STEP RATE CODE IN HEX:	1
PARKING CYLINDER IN HEX	268 (134)

It takes approximately 1 hour and 40 minutes to FASTPREP a ten megabyte Bernoulli cartridge. By comparison, it takes approximately 3 hours and 45 minutes if you use PREP instead of FASTPREP. After completion of FASTPREP, you are ready to partition the cartridge.

Zenith has a limitation of the size of ZDOS and CP/M partitions. Each ZDOS or CP/M partition is limited to a maximum size of 7.9 megabytes. MSDOS 2.0 and later versions will support partitions of up to 32 megabytes. If the Gemini board is used, Gemini partitions are limited to a maximum size of 17 megabytes per partition.

If you have a Gemini board, several comments are appropriate here. First, when you are creating a Gemini partition by running PART in the Z-100 mode, you must name that partition "Gemini" and the response to the "Operating System Name" should be "Drive". The Gemini board supports a maximum of two physical drives and one Gemini partition may exist on each physical drive.

Insert the disk containing PART.COM and execute the program by typing:

```
PART
```

The instructions for running PART .COM are contained in the MSDOS documentation. The only deviation from the documentation should be for those of you creating a Gemini partition, as explained in the previous paragraph. You may create Z-100, Gemini and CP/M partitions on a single cartridge. You can create up to four Z-100 partitions and two CP/M-85 partitions. If you are going to use two Bernoulli drives, then these are the total number of partitions for both drives. The limitations on Gemini partitions has been previously stated. After PART has finished, you must reboot your computer as PART is destructive to portions of the operating system resident in memory.

After the Bernoulli cartridge has been partitioned and the computer rebooted, you must assign the partitions and format them. Assigning drives and formatting them is done using three different operating systems. Reboot the computer by using either ZDOS/MSDOS, CP/M-85 or H/Z-150 PCDOS; the choice depends upon which partition is to be assigned and formatted.

Assigning a partition to a logical drive letter is done by using ASGNPART for MSDOS 3.0 or ASSIGN for MSDOS 2.0, ZDOS or CP/M. The instructions for using these commands are in the documentation on the appropriate operating system. It is not necessary to assign the partitions for the Gemini as this is done automatically. The first Bernoulli (or only Bernoulli recognized by the controller) is assigned as logical drive C and the second (if applicable) is assigned as logical drive D. For the Z-100 under MSDOS, Versions 2 and

3, the logical drives available for assignment are E, F, G and H. Drives C and D are reserved for eight inch drives. Under CP/M-85, the logical drives that may be assigned are C and D. I did not have the opportunity to experiment, but the documentation indicates that if there are eight inch drives on the system, then drives C and D are assigned to them and the logical drives available for assignment are E and F.

Once the partitions have been assigned to a logical drive, they may be formatted exactly as you would format a floppy disk. Whenever the computer is rebooted, the Bernoulli partitions have to be reassigned. You should include the assign commands in your AUTOEXEC.BAT file (for ZDOS/MSDOS and H/Z-150 PC-DOS). It is possible to create a bootable partition and boot from it, although I did not do this. In this case, the booted partition does not have to be assigned. The remaining partitions must still be assigned.

This completes the initialization of the Bernoulli cartridge and it is ready for use.

Problems

Only one limitation was found in operating with the Bernoulli Box. When booting up in the Gemini mode, you should notice the following brief (and I mean brief) messages displayed after you have entered I (for IBM mode) and before the Z-150 monitor boot message:

```
GEMINI PARTITION OF WINCHESTER 0 INSTALLED AS IBM WINCHESTER 0
```

and if you have two Bernoulli drives, you should also see

```
GEMINI PARTITION OF WINCHESTER 1 INSTALLED AS IBM WINCHESTER 1
```

If you do not see these messages, you will see one telling you that no Winchester drives are installed. The problem lies with the Bernoulli Box. If no disk access has been accomplished for a given period of time, the drive speed is reduced. The software in the Gemini EPROMS do not allow sufficient time for the drives to come up to speed and respond with the proper "ready" signal. To solve the problem, you should manually "spin down" the drive by pressing the button on the Bernoulli. When the green LED is out, the drive has stopped. Then turn the lever on the drive counterclockwise to release the cartridge and then clockwise to relock the cartridge. The drive will begin to come up to speed as indicated by a flashing green LED. When the drive is at full speed, the LED will stop flashing and remain green. There is no indication on the LED when the drive is at reduced speed. It will take practice to synchronize rebooting your computer and your Bernoulli. I found that if I had the Gemini boot menu displayed on the computer screen, I could reboot the Bernoulli, wait three seconds and then boot the computer by pressing I (for IBM mode) and be successful nearly 100% of the time. This problem does not exist for the Z-100 mode (DOS or CP/M), nor does it exist during normal operation in the Gemini mode; only when booting the Gemini mode.

Conclusion

I hope that this article has been instructive. I have made every effort to be accurate in my statements and have verified them by repetitive testing. I realize that it is not possible to cover every detail, but I believe that the documentation that accompanies both disk controllers will clarify any points left unanswered.

Vendors Mentioned

Altex Electronics, Inc.
10731 Gulfdale
San Antonio, TX 78216
(512) 340-2409

C.D.R. Systems, Inc.
7171 Ronson Road
San Diego, CA 92111

Gemini Technology Corp.
13227 North East 20th Street
Bellevue, WA 98005
(206) 747-5267

IOMEGA Corporation
1821 West 4000 South
Roy, Utah 84067

Lindley Systems
21 Hancock Street
Bedford, MA 01730-1613

Scottie Systems
2667 Cropley Avenue #123
San Jose, CA 95132
(408) 259-6226

Zenith Data Systems
Attn: Jane Eaton
Hilltop Road
St. Joseph, MI 49085

*

Continued from Page 22

pleted and the memory settings are correct, re-boot the computer. As it boots, you'll see numbers on your screen incrementing upward as the AST memory board is tested. Once the memory test is complete, the machine finishes the boot process. Assuming everything tests out, you are ready to go. Remember to tighten the screws for the machine cover and Happy Computing!

*

Graphics Printer or Epson FX Part 2

Controlling Your Printer

John A. Day
5 rue Sauer
77500 Chelles, FRANCE

Controlling Your Printer

The first article in this series went into the general differences between FX and GP printers, which are the two standards you will find together at the flip of a switch, on most small matrix printers. Now, we'll get down to the detailed control codes. Basic control is covered by ASCII (American Standard Code for Information Interchange). We've already mentioned the 32 codes set aside for this purpose, and which you'll find plus/minus a few on most any printer. Here again, the FX and GP standards are not identical. Both cover:

7 BEL	Beep about 0.1 sec.
9 HT	Horizontal Tab
10 LF	Line Feed
11 VT	Vertical Tab
12 FF	Form Feed to top of next page
13 CR	Carriage Return
14 SO	Shift Out
15 SI	Shift In (condense to 60% width)
18 DC2	Device Control
20 DC4	Device Control
24 CAN	Cancel Contents of Print Buffer
27 ESC	Escape Sequence

24 CAN serves no useful purpose, and you won't use *11 VT, 14 SO* and *20 DC4* either. On Heath/Zenith computers, *CR* means return the print head to the left margin without paper advance, and *LF* means advance the platen to the next line. Both codes together are needed to change lines. The two shift codes do not shift between upper- and lowercase. *SO* sets double-width characters, but there are better ways of selecting this. *SI* is a good way to select condensed printing, which gives characters about 60% of their normal width. "Device Control" covers four codes from 17 thru 20. *DC1* and *DC3* can put an FX off- and on-line if the internal switches are set a certain way and if a particular pin on the parallel port is high. I have no idea what use this could be. *DC2* cancels condensed printing, and returns the printer to normal width (or double width if selected). *DC4* cancels *SO*, which we're not using. All these codes are meant to be interspersed with the text to print, and the printer handles them in the proper order.

The FX is a backspacing printer, while the GP is not. This means that the FX has two extra ASCII codes:

8 BS	Backspace One Character width
127 DEL	Delete Last Character in Buffer

Backspacing is a very slow operation, which can be used for overprinting. The head must be stopped, backed up 3/4" because of the 3/8" stopping distance, and then restarted to overprint the next character on the fly. If you have two or three characters to overprint on a line, the printer will spend more time stopping and starting than printing. A full line overprint is much faster: Set up a new line with just the characters to overprint, issue a *CR* without *LF* to come back to the left margin, and print the line. The printer will do the whole lot on the fly. If you're using a GP, you'll have to do it this efficient way, and I'm all for efficiency. You will notice that the TSR program given in the previous article backspaces, but it can't avoid this because printer drivers work character-by-character and not line-by-line.

So far, it's easy, isn't it? Well, now things get complicated. ASCII defines code *27 ESC*, as the start of an escape sequence. The printer designer can put anything he likes after an escape code, since ASCII leaves him a free field. Two conflicting tendencies prevail:

- If you have a minority position on the market, you try to include everybody else's escape sequences in your printer, so that you can pick up extra sales.
- If you are in a majority position, you slip in a few jokers so that rival printers won't run as well as replacements.

The result is that there are often three or four ways of doing something, corresponding to ideas copied from three or four sources, and there are some very unsubtle differences to trap the unwary.

There are four main standards for escape sequences, which we will call 'A' (ANSI), 'D' (Daisywheel), 'F' (FX), and 'G' (GP). The ANSI national standard is a cumbersome standard, difficult to decode by the printer firmware, and hardly anyone uses it. 'A' sequences usually start with *ESC [*, continue with a numeric value in standard ASCII code (over 1, 2, or 3 bytes), and close with a character defining the operation. This is backwards with respect to the other standards, which usually read *ESC*, a character defining the operation, and if necessary a binary byte with a numeric value. The main advantage of 'A' codes is their comprehensive nature; they cover all sorts of I/O, including the screen controls you get in the ANSI.SYS driver. The 'D' codes were developed for word processing printers. Type styles can't be set by software, as they entail stopping, manually changing the daisywheel and restarting, so the 'D' standard doesn't include width and style commands. It does, however, contain useful instructions for justifying text on a line, and modern FX and GP emulations often add a few 'D' codes for efficient word processing.

Small matrix printers almost all offer a choice between the 'F' and 'G' code sets. I'll just sum up rapidly the main differences between the two, before going into the best use of both codes. The following 'F' instructions were not included in the 'G' set:

- Anything that moves backwards, that is, backspacing and reverse paper feeds;
- Elite (12 cpi) printing;
- Italics
- Single instruction print mode selection (ESC !)
- Downloading user-defined fonts
- Proportional spacing
- Bits and pieces like running at half-speed, setting margins and vertical tabs.

There are also the jokers, which give different results on the two instruction sets:

- Fine feed instructions give an implied carriage return in the 'G' set;
- Line spacing commands are subtly different, in such a way that non-IBM software will tend to stick at 6 lpi on a GP, and IBM software will do the same on an FX.

As you can see, there is much more available on the FX emulation, which probably explains why it is still going extremely strong even though it doesn't use the IBM PC character set. Most small printers have both sets of characters and codes available, with a switch to swap between the two. My own printer, a Centronics PS 220, is always in Epson mode, but with a nearly full set of GP characters as a user-defined font. I like the best of both worlds... Some manufacturers provide a software switch to move dynamically between 'F' and 'G' modes; others point out that a) Epson has not included anything in their standard for IBM emulation (although they have taken recently to including an IBM character font), and b) IBM has not included anything in their standard for Epson emulation, so Beware of printers such as the Centronics 'Horizon', which combine both modes at the same time. The 'Horizon' is my favorite printer; it does everything, and I started in on software for the PS 220 to try to get this last to do as much as the other. The 'Horizon', however, was always full of glitches you had to program around, the worst being a propensity of the narrow platen version to burn out while printing French text in 12 cpi mode. The wide platen version, which equips my group's admin computers, won't be retired until it falls apart. But you'll only find 'Horizons' in surplus sales now. Centronics must have had too many headaches trying to put everything in one box, and their new ranges were 'F' or 'G', with hardware switches, as on most other makes. These switches are never easy to get at on any printer. The following discussion will let you weigh up the pros and cons for each emulation, and then set switches once and for all according to which codes you need the most.

Downloading IBM fonts into an Epson will be for one of the closing chapters, when I will ask Jim Buszkiewicz to put two binary files for this onto the bulletin board, together with the Assembler source. For the moment, I shall hope you've chosen 'Epson' emulation and go into how to use it to the best. For each command, I shall indicate at the same time if it doesn't work on a GP.

First of all, draft characters and pitch. We've seen how the usual head speed gives dots spaced at 1/60" with stagger. The head can be slowed down while keeping the same frequency for energizing the pin solenoids, which squashes the letter horizontally:

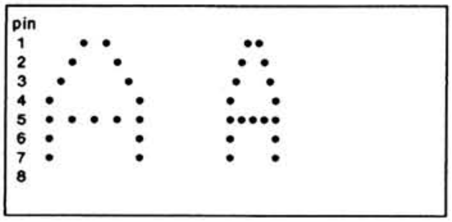


Figure 2-1
Condensing a Character by Slowing the Head

Running at 83%, 60% and 50% of standard speed gives 12, 17 and 20 cpi, respectively. You can mix sizes on a single line, but the printer is unlikely to change pitches without stopping the head, backing up and restarting. For rosters and charts where I have several inserts at 17 cpi in a 12 cpi line, I always print the line in two goes, as for backspacing; a first pass prints everything for 12 cpi, leaving gaps for the parts in 17 cpi, and a second pass fills in the gaps.

The head can't be speeded up to print wider, but dots can be printed twice:

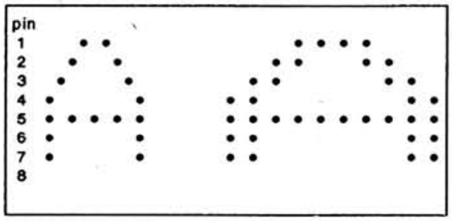


Figure 2-2
Enlarging a Character by Printing Double

The pitch between dots stays at the standard value, which doubles the character width while maintaining the standard firing rate. The basic 10 cpi doubles to 5 cpi, while 12 and 17 cpi double to 6 and 8.5 cpi, respectively.

To enlarge characters to double width, send *ESC "W" 1*, followed by *ESC "W" 0* to come back to normal pitch. 12 cpi, in 'F' only, is Elite mode, selected by *ESC "M"* and cleared by *ESC "P"*. Condensed characters, at 60% normal width, are selected by *SI* and cleared by *DC2*. You can mix two selections: 8.5 cpi is either *SI ESC "W" 1* or *ESC "W" 1 SI*. Not all printers allow *ESC "M" ESC "W" 1*, which selects 20 cpi on my PS 220. In BASIC, define ASCII control codes at the beginning of your program, together with binary "0" and "1":

```
Z$ = CHR$(0): 0$ = CHR$(1)
SI$ = CHR$(15): DC2$ = CHR$(18): E$ =
     CHR$(27)
```

You can then switch easily between pitches:

```
LPRINT SI$;E$"W"O$"Chapter 1"DC2$;
     E$"W"Z$;
```

will print Chapter 1 at 8.5 cpi, and revert to standard pitch, but see the chapter on "Interference from BASIC" further on. Don't forget the closing semicolon if you don't want a new line after a control sequence. If you use the same pitch changes at several points in your program, define a simple variable for the control sequence:

```
CPI85$ = SI$ + E$ "W" + O$
CPI10$ = DC2$ + E$ + "W" + Z$
```

dBASE lets you handle control sequences nearly the same way, except you omit the $ and you can't use a 0.

Except for *ESC "M"* and *ESC "P"*, the previous works with 'F' or 'G' instructions. A nicer way to set pitches is with the 'F' instruction *ESC "!"* n. *n* is a combination of values that lets you set or clear several modes simultaneously. "1" is Elite 12 cpi, "4" is condensed and "32" is enlarged. Combining gives, for example, "33" (1 + 32) for Elite enlarged at 6 cpi, or "36" (4 + 32) for the BASIC example above:

```
LPRINT E$"!"CHR$(36)"Chapter 1"E$"!"Z$;
```

Anything that isn't explicitly set is cleared, and this sequence may also control italics, underlining, and other features, so you must know exactly what print style is already in use if you don't want to clear something inadvertently. If you want to set or clear a single feature without modifying the others, use the codes in the other paragraph.

Now for other horizontal settings. If you have a 10" platen, then the standard print width is 8" for 80 pica characters. With a wider platen and standard 9-1/2" fanfold stock, you will find certain program files running off the right of the paper when you print them with the DOS type instruction. *ESC "Q"* n sets a right margin at column n in whatever pitch you are using; *ESC "l"* n (lowercase L) does the samė for the left margin; both are 'F' only. Thus,

```
LPRINT E$"!"O$;E$"l"CHR$(12);
     E$"Q"CHR$(96);
```

will set a left margin at 1" to allow for filing holes, and a right margin at 8", all subsequent printing being in 12 cpi with 85 characters per line; the *ESC "!" 1* at the beginning means that the margins are set in 12 cpi column values. Any line that is longer than 85 characters will be cut at the right margin, and continued on the next line. You can also use this with dBASE before a list to print instruction, but you can't clear a left margin in dBASE, because it hasn't got the zero for *ESC "l" 0*. If you change print pitch after setting the margins, they should stay where they were in inches from the left, as the printer stocks them as absolute positions and not as printer columns.

Tabs are useful. When you power up your printer, you have tabs set every 0.8". 'F' pitch changes don't alter tab positions, which are stored as absolute values. Condensed printing in 'G' mode will move the tabs, as they are stored in the form of column numbers. Each HT code slews the printhead forward to the next tab. This is the one example where the printhead can change speed without stopping: better-quality printers slew faster than for printing, and slow down in time to switch to Pica printing mode (but for condensed printing, may well stop and start). Tabs let you quickly align text on a particular column, and are economical on disk space, since one tab character replaces up to eight spaces. If you are mixing different pitches in 'F' mode, tabbing is about the only way of lining up columns. You may also set your own tabs, up to 32 absolute positions in 'F' mode or up to 28 column numbers in 'G' mode. The control sequence is *ESC "D" n1 n2 ... O*; omit the "O" if you give the maximum number of tabs. *n1, n2* ... are the tab positions, in ascending order, given as column numbers in the current print width. As an example, you can set a first tab at column 50, and a second at column 65:

```
LPRINT E$"D"CHR$(50);CHR$(65);Z$;
```

For frequent tabbing, define a tab character:

```
HT$ = CHR$(9)
```

and, for instance, you can start an address in column 50:

```
LPRINT HT$"Mr. So-and-so."
```

Of course, you can tab directly in BASIC with the *TAB(n)* function, but this won't handle pitch changes properly.

Before printing any Assembler output, I give the sequence:

```
LPRINT E$"!"CHR$(4);E$"D"CHR$(8);
     CHR$(16);CHR$(24).....;CHR$(104);
     CHR$(112);CHR$(120);Z$;
```

This selects condensed 17 cpi, to give me 132 characters on an 8" form, and sets the tabs the same way as MASM uses them. This way, I can use tabs in my Assembly programs for op codes and operands, and quickly line up my comments in column 41 where I like them. Similar code will let you print BASIC compiler listings in 12 cpi, 96 columns per page, which gives you a good 80 columns of program text, plus the compiler narrative on the left. I was brought up on structured programming, so I indent my BASIC code automatically: four spaces for a first-level indent, one tab for the second level, then a tab and four spaces and so on.

While timing my printer with a stopwatch to check some of the figures in these articles, I was very surprised to discover that as soon as I use tab character, it goes into one-way left-to-right printing. I wonder whether that is the reason behind the smoked perspex cover which barely lets you see where the print head is? The Centronics 'Horizon' has clear perspex covers, and prints at high speed with or without tabs.

The next article will cover useful control sequences for vertical spacing and special effects; choosing character sets will be in Part 4. ✱

DOS And Unix

Part I

Tom Bing
2755 Carolyn Drive
Smyrna, GA 30080

Differences And Similarities

PC users are seeing more about the Unix operating system nowadays. The 'big iron' 286 and 386 machines needed to run Unix efficiently on a desktop are becoming more common, and good articles on the subject have already appeared in REMark and other leading PC publications.

The objectives of this article are to show the DOS user around Unix and to use his or her DOS knowledge as a springboard to achieve a basic understanding of important Unix features and commands. "But why?", some may ask. "I've got my hands full getting DOS to do all I want. I've spent a lot on DOS software. I know I can buy Xenix or other Unix flavors, but it's still a bit expensive for me, and besides, I just don't have enough horsepower in my Z-150 to run it. I've heard it has a lot of nifty utilities, though." Rejoice and read on! This article was written with just that attitude in mind. When this article series is over, I think I will have made two points:

1. The learning curve for Unix isn't that steep, especially in light of the power of the new knowledge you gain.
2. There are affordable tool sets to bring the most useful Unix utilities to your PC without the need for gobs of RAM (or money). And you don't have to junk your DOS programs either.

This series will be of most help to someone who has to alternate between a Unix system at work and a DOS machine at home, or someone who is "bumping into walls" with DOS. Those of us who use DOS for a while wind up buying utility packages to get around its limitations. I use Lotus Metro at the office for this reason. But I found a package called MKS Toolkit which has a pot full of programs that handle the analysis and data transformation jobs that I do all day. Apart from the modest purchase price, my only investment was to study the excellent Toolkit manual and try some of the different commands. In the process, I acquired a better understanding of both DOS and Unix. You see, MKS Toolkit includes the more common Unix utilities on a set of DOS disks. We'll explore why you would want to have it, in the next article. It can be set up in four different ways, ranging from very DOS-like to very Unix-like. It offers the maximum payback in speed and power in the Unix-like configurations. That's why this article will dwell on the comparison of DOS to Unix.

MS-DOS Version 3.x ('DOS') and Unix System V are the basis of the discussions below. Very little would change, however, if somewhat earlier versions were used. I will make note of particular differences between Unix V and Xenix, for instance, where I know about them. MKS Toolkit is designed for a high degree of compatibility with Unix System V.

First, the basics, in table form:

	Unix	DOS
Type of OS	Multi-user Multi-tasking	Single-user Single-tasking
CPU Family Used	Many vendors	Intel 80×86
Typical User Terminal	Any of several ASCII serial terminals (Heath H-19, DEC VT-100, etc.)	PC Console and keyboard
First Version	For DEC PDP-7, 1969	For IBM PC (Model 5150), 1981

Unix and DOS (from version 2.0 on) share some basic features. The multi-level directory tree, redirection of input and output (>, >>, and <), the 'pipe' (|) to allow one program's output to become another program's input, are all DOS features borrowed from Unix. The ANSI.SYS file in DOS is an example of what the trade calls an 'installable device driver'; it is a piece of software designed to 'know' the special characteristics of an I/O device (the PC console, in this case), so that other software (editors and spreadsheets, for instance) won't have to. (In this case, it's probably more accurate to say that the original end-user software is 'looking for' an ANSI-standard terminal, and ANSI.SYS provides the required translation between the escape sequences for an ANSI terminal and those commands needed by the PC display hardware.)

The Unix world had to support various terminals and printers early on, and Unix even in its infancy was being ported to other machines besides the PDP-7. This diversity of hardware led to the idea of isolating the machine-dependent part of Unix in special files and tables. Unix also has a set of tables called 'termcap' (Xenix) or 'terminfo' (System V), which describes dozens of terminal types and their features. Any user's terminal type only has to be given once in his '.profile' file, the Unix counterpart of AUTOEXEC.BAT. This way, word processors and other programs that use full-screen features don't have to have a lot of different terminal specs coded into them.

DOS and Unix do a lot of the same jobs slightly differently, though. As an example, the full 'path name' of a text file in DOS might be 'C:\DOCS\LISTS\PHONE .LIS'. In Unix, the same type of file might be '/docs/lists/phone.lis'. Unix uses '/' instead of DOS's '\' to separate directory names in a path. It would be possible to have a totally different Unix file on the same machine named '/docs/lists/PHONE.LIS', since upper and lower case names are distinct in Unix. If Unix uses '\' as we've seen, what do we use to identify options to a Unix command? Glad you asked. The hyphen is to Unix what '/' is to DOS. For example, the wide directory listing (DIR /W in DOS) has as its closest Unix equivalent 'ls -C'. Unlike '/', the hyphen can usually be followed by more than one option; 'ls -lt' in Unix produces a 'long' or detailed (-l) directory listing sorted in time (-t) order, newest files first. While 'phone .lis' is a valid Unix file name, 'phone.listing' is also; Unix file names can be up to 14 characters long and there are no other length limits for parts of the file name (no 3-character extensions, for instance). The only character which file names cannot contain is '/', since it is the directory separator. However, it is best not to use '[', '?', and a few other punctuation marks, since they have special meanings in the Unix shell.

Now let's look at common DOS commands and their Unix counterparts. DOS commands are shown below in UPPERCASE; Unix commands are given in single quotes in lowercase — see CD below.

CD/CHDIR: 'cd' in Unix. Note that CD by itself reports the name of the current directory and doesn't move you anywhere else; 'cd' by itself in Unix changes directories to the user's HOME directory — the one you are in when you log in. To find what Unix directory you're in, type 'pwd'. The thing to remember is that Unix 'cd' always moves you somewhere else in the directory tree unless the requested directory is your current one or is invalid (nonexistent) or 'off limits' to you. Unix directories and files have a set of permissions that govern which users can access them. These permissions can be changed by the owner of the directory or file or by the 'superuser' of the Unix system. Notice that if you are 'on' DOS drive C, and you type 'CD A:\SUB1', you don't move to the A disk. This CD command simply means that later DOS commands like 'DIR A:' will refer to the SUB1 directory on drive A. In a Unix system where '/f' is the floppy disk drive, the command cd '/f/sub1' will move you to that directory. This example shows the 'cleanness' of the Unix design; one command does one task, or only one kind of task. Some earlier Unix systems had 'chdir' as an alternate name for this command also.

If we use CD in a DOS batch file to change directories, the current directory after the batch file run will be the last one we moved to using CD. In contrast, a Unix shell script may include 'cd' commands, but when the script completes, the current directory will be the one in which the script run began. The current directory is part of the 'environment' of a Unix process. The script file run is done as a 'child process' by the Unix shell, and it is a basic principle in Unix that a child process cannot change the environment of its parent. Of course, while the 'child' is running, the 'cd' commands really work; their effect is temporary and does not outlast the script run.

CHKDSK: 'du' in Unix reports disk space used in the directory specified, including subdirectories. The number returned is in 'blocks' or sectors. Sector size may vary between Unix systems depending on the size of hard disk used. The 'df' command reports the number of unused blocks remaining. The Unix 'fsck' (file system check) command is used to report and fix problems with the Unix file system (such as after a power outage) as DOS 'CHKDSK /F' does.

COPY: 'cp' in Unix. Like COPY, 'cp' will not allow the input and output files to be the same and will allow the last entry on the command line to be a directory. Unlike COPY, we can copy files 'one', 'two' and 'three' to directory 'sub1' by typing:

```
cp one two three sub1
```

If these three files were the only ones in the current directory, 'cp * sub1' would have the same effect. The /A (ASCII) and /B (binary) options have no counterpart in 'cp', because 'cp' finds the end of a file by the number of bytes in the file, not by the presence of a Control-Z character.

The COPY /V (verify) option has no 'cp' counterpart. To check and see that a copy was performed without error, the 'diff' command could be used to compare source and destination files after the copy is done.

'cp' cannot concatenate files as 'COPY A+B+C D' would. The Unix 'cat' command does this (see TYPE below).

DATE: The Unix 'date' command by itself simply prints the day of week, date and time, like this:

```
Fri Jul 15 13:45:53 EDT 1988
```

It does not prompt for a new date. If the superuser types 'date' with a new date and time following it in the right format, the system will reset the date. Ordinary users can get date and time information back in different formats by using certain options. For instance, the command

```
date '+%y%m%d'
```

returns 880715 for the date above. This is useful for creating file names keyed to dates.

DEL/ERASE: 'rm' in Unix. This command must be used with some caution, because few safeguards for its use are built into 'rm' itself. For instance, DOS will ask us "Are you sure?" If we type 'DEL *.*', but for the far more destructive 'rm -r *' in Unix, which means simply "let there be nothing" — blow away all the files in this directory and all subdirectories — it starts deleting files and removing directories right after you hit RETURN. There are some safety nets, though. Since 'rm' is a program on disk in Unix (DEL is built into DOS), the savvy superuser can fix it so that 'rm' can only be run with the '-i' (interactive) option, which will prompt the user with the name of the file to be removed and a question mark. Only if the user enters 'y' will the file be deleted. Individual users can protect themselves by creating an 'alias' to get the same result. Even in its raw form, 'rm -r' will not allow you to remove the current directory, which would be like sawing off the tree limb you are sitting on.

DIR: 'ls' in Unix. 'ls' by itself produces one long column of file and subdirectory names. There are options to give multiple columns of names (-C), reverse the sort order of the list (-r), show '/' at the end of directory names (-p) and so on. 'ls' will show all the file and directory names one level below the current directory unless the '-d' option is used. For example, the command 'ls -prd a*' would produce the following result for file names 'alpha.doc', 'albert.doc', 'adams', and subdirectory 'apartment':

```
apartment/
alpha.doc
albert.doc
adams
```

The beauty of 'ls' is that the output can be used directly, without a lot of finagling, as input to other programs. For instance, the command 'ls -d a* | wc -l' (which means 'make a list of all files and directories beginning with 'a' and count the number of lines in the list') would produce the output '4', in this case. With DOS DIR, we would have gotten 'ALBERT DOC' as one file name, along with date, time, and other stuff, which would have required editing to ALBERT.DOC before a program could recognize it as a file name.

ECHO: 'echo' in Unix. 'echo' does one thing — it echoes whatever comes after it to the 'standard output' — usually the terminal screen, but output can be redirected with '>' or '|'. For instance, typing 'echo on' simply causes the word 'on' to print on the next line. 'echo on' in DOS would normally be used in a batch file to turn on the display of commands as they were executed, assuming 'echo off' had been used in the file before. If a Unix user wanted commands in a shell script (= batch file) to display as they executed, the command 'set -]x' would be inserted right before the first line to be echoed. Turning this display off is done with 'set -', but usually isn't needed, because this is the default condition which is restored automatically after a shell script stops running.

FIND: 'grep' or 'egrep' in Unix. 'grep' is short for 'global regular expression print'. The 'global' part refers to the 'print'; it means "print EVERY instance of this regular expression that you find". For instance,

```
grep -i Smith phone.lis
```

would print every line of the phone .lis file that had the word "Smith" on it (or "smith", or "smITH"; the -i means 'ignore case'). The term 'regular expression' in Unix refers to a pattern that can be matched by one or more strings of characters. For instance, the command

```
grep [A-Z][0-9] phone.lis
```

means "print every line in phone.lis where a capital letter comes right before a number". 'egrep' is 'extended grep', and can use a more broadly defined set of regular expressions than 'grep' does.

There is also a Unix 'find' command. Its job is to find files in the directory tree that match conditions specified by the user, such as all files changed within the last 15 days. The file names found can then be printed or fed to another program, such as one that copies the files to a floppy disk.

MD/MKDIR: 'mkdir' in Unix. DOS and Unix match closely here, except that 'md' is not usually allowed as an alias unless the user explicitly sets it up that way. Since Unix directories, as well as files have permissions, it's possible that a Unix user will be allowed to read all the contents of a directory, but won't be allowed to create subdirectories under it.

MORE: 'more' on Xenix and Berkeley Unix; 'pg' in System V. These commands allow the user to view a text file a screenful at a time. MORE in DOS requires the '<' symbol when input is from a file (MORE < FILE.TXT). The Unix and Xenix commands are simply 'more file.txt' and 'pg file.txt'. 'pg' has so many options for seeking backward and forward through the file and executing other system commands during its run that it amounts to a read-only text editor.

PATH: Unix 'PATH' is what we call a shell variable; shell variables are special identifiers whose values are known and used by the shell and other programs. Like DOS PATH, the Unix variable specifies the directories to be searched to find the command whose name is entered by the user. While a DOS PATH value might be "C:\DOS;C:\;C:\UTILS;" with semicolons between the directory names, the Unix PATH uses colon delimiters, like "/bin:/usr/bin:/usr/local:". The current directory, if used in PATH, appears in the middle of a list as "::" or on the end as ":". The two commands below could be issued at the operating system prompts in DOS and Unix with similar results.

```
DOS:  SET PATH = C:\DOS;C:\;C:\UTILS;
Unix: PATH=/bin:/usr/bin:/usr/local:
```

DOS can also set PATH with a resident command (PATH C:\DOS;C:\; C:\UTILS;). Unix only uses the one method above. Unix will allow a directory to be 'tacked on' after the original PATH assignment: PATH=${PATH}/newdir: is an example.

PRINT: The Unix 'lpr' print spooler is used to send files to the printer and is often used with the 'pr' print formatter. 'pr' can add title lines, fold lines at a given column, expand tabs, and so on. There is a Unix 'print' command, but it is really 'echo' with some added features.

PROMPT: In Unix, the prompt is set by assigning values to shell variables PS1 (the primary prompt, usually '$') and PS2 (the secondary prompt). PS2 is usually set to ">" and is displayed when a user's command exceeds one line of input. Certain punctuation in a Unix command line, such as '\' or an unmatched single quote, will allow a command to be continued on the next line.

RD/RMDIR: Unix 'rmdir' is quite similar in that a directory has to be empty before it can be removed, and neither DOS nor Unix allows removal of the current directory or its parents. Unix 'rmdir' allows removal of several directories with one command using wild cards. The command

```
rmdir a*
```

removes all empty directories in the current directory whose names start with 'a'.

RENAME: 'mv' in Unix. Neither RENAME nor 'mv' will put an existing hard disk file on a floppy, or vice versa, since this is really a file copy. The commands below work alike:

```
DOS:  RENAME CAPULET MONTAGUE
Unix: mv capulet montague
```

RENAME can include path names only in the first file name, to place it in the current directory under its new name (such as RENAME C:\DOCS\DEMO.DOC MEMO1.DOC). 'mv' can have path names in both file names. 'mv' can move several files to another directory, such as:

```
mv list1 list2 list3 /usr/tom/dir1
```

In fact, if 'mv' has three or more names after it, the last name has to be a directory, and the other names must be files. Wildcards may be used. For instance,

```
mv a* docs
```

moves all files in the current directory whose names start with 'a' to the subdirectory 'docs'. If a target file name already exists, it will be destroyed by 'mv' if the user has write permission on it.

TIME: Unix 'date' reports the time, as well as the date. There is a Unix 'time' command, but its job is to report the CPU and elapsed time taken to run commands.

TYPE: 'cat' in Unix. A useful 'cat' option is -v, which makes control characters in source files visible. For instance, if a file named 'notes' had an escape character in the text, the 'cat -v notes' command would display the escape as ^]. 'cat' is used to concatenate several files into one. For instance,

```
cat d* > ../d_files
```

will copy all files in the current directory whose names start with 'd', one after another, into a file named d_files in the parent directory.

Now what? As we've scanned these commands, you may have noticed some desirable new Unix features and some DOS capabilities you don't want to lose. The MKS Toolkit allows you to have some Unix functions and features on a DOS PC. At the same time, you can still run DOS applications programs. In the next article, we'll go over some of the more useful utilities included in the MKS Toolkit. ✱

Continued from Page 14

location, and save input data. As in The_Sky, up to five locations can be added. You must also come here to exit to DOS.

The following is a listing of their 'Special' keys: When in doubt, hit the F1 key. F1 in any non-graphic screen will give help. F2 is the edit key. Saves retyping. F9 switches from graphics to text type display to output the screen to a non-graphics printer. F10 starts computations. This is a rough one to remember. When I enter data, I hit the enter key but nothing happens.

Positional comes up with 6 functions: Planets, Sun, Moon, New and Full moon, Jovian Moons, and Eclipses. The first four are current data positions in right ascension and declination. The last two are graphic. Jovian Moons shows the moons of Jupiter on a plane view and as seen from a telescope. The plane view angle can be changed, and the time interval can be increased or decreased to show the moons orbiting around Jupiter.

Eclipses, will show you the time and date of the next lunar or solar eclipse, with an actual demonstration of what it will look like at your latitude. Pressing the space bar causes the sun or moon to move into the eclipse, another spacebar moves them out, and one more starts calculations for the next eclipse. These are good OOOH! and AHHHH! programs.

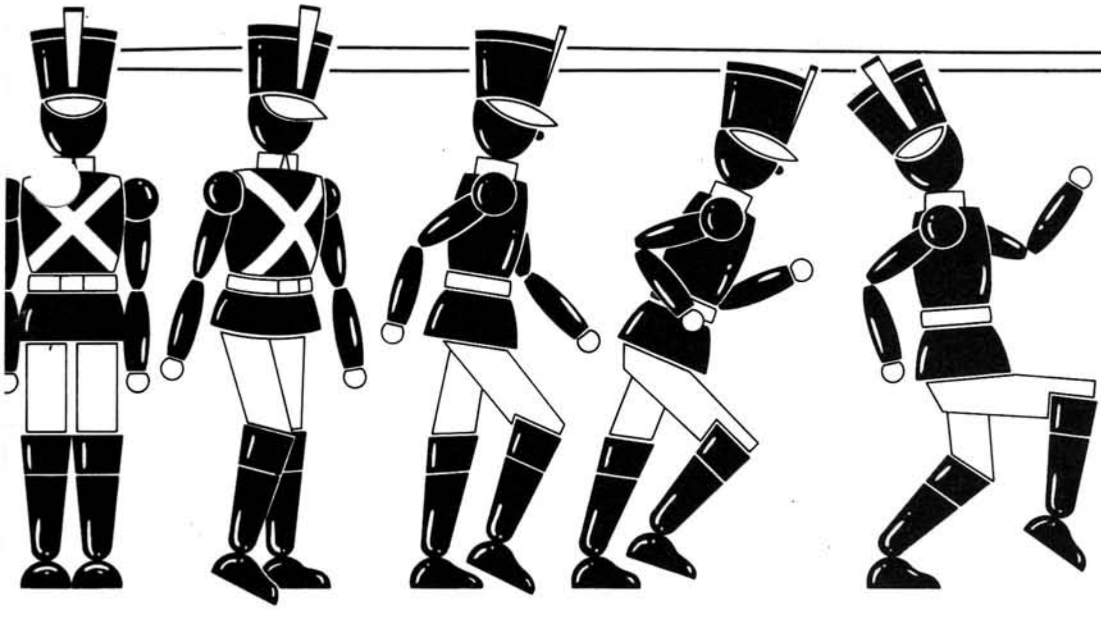
Conversions is self-explanatory with selections being Time, Degrees to and from decimal, Horizon, Ecliptic, and Galactic to and from Equatorial.

Miscellaneous lists Ephemeris, Nutation, Precession, Set-Rise, Time Display, Monthly Calendar, and Annual Calendar. They can all be sent to a printer.

Options list Color, Graphics Mode, Printer, ESC Sequences, Form Feed, and Version 1.00. Color mode support is the same as The_Sky.

I just received another shareware program on stars, and it has also joined the others in the trash can. The_Sky and Sky_Calc are still the best around, and I'm not in a hurry to replace my telescope anymore.

For further information call or write:
COMPUTER ASSIST
1122 13th Street
GOLDEN, CO 80401
(303)-279-8073 ✱



The Battle of the Boards

Another Look at the AMI ZX-386 and AOX Z-Master 386 Upgrades

Pat Swayne
HUG Software Engineer

The AMI ZX-386 and the AOX Z-Master 386 are 80386 upgrades for the H/Z-241 and H/Z-248 (80286 based) computer systems. I reviewed the ZX-386 in the January issue of REMark, and the Z-Master 386 in the June issue (both this year). Since writing those articles, I have continued to "play" with both boards, discovering additional strengths and weaknesses, and I have gathered additional information from their manufacturers. Because of this, I have written this additional article on the boards.

A Little Editorial

Another reason for this article is that I hope that if you have a Z-248 type computer and you are thinking of upgrading to an 80386 machine, that you will seriously consider getting one of these boards. Those of you who have been with HUG for several years will remember that back when the H-89 computer was in its heyday, there were at least a half dozen companies producing add-on boards and other products of every description for it. The H-89 has all but faded away, and so, it seems, have third party add-on products for Heath and Zenith computers. You may be thinking that since today's Heath and Zenith computers are "clones" of everybody else's computers, that there is not a market for add-ons specifically for our computers. But the ZX-386 and the Z-Master 386 are proof that there really is such a market. If the companies that produced these boards see enough support for their products, perhaps we will see additional upgrade products from them. How about a 33-MHz 80386 upgrade for the original 16-MHz Z-386? Or (looking a little farther down the road), how about an 80486 upgrade?

The New ZX-386 ROM

In my June article on the Z-Master 386, I mentioned that since my January article on the ZX-386, AMI had provided a new ROM that supports ROM shadowing. As it turns out, I am only one of a few (as of this writing) that have the new ROM, and AMI is still shipping the old one that contains test utilities instead of the new Extended Setup feature I had mentioned. Since some time will have passed before you can read this, you should check with AMI to see what kind of ROM is available for the ZX-386. The test utilities are kind of nice to have available, and as you will see later in this article, you really do not need a special ROM to have the benefits of ROM shadowing.

The Z-Master 386 Speed Utility

I also mentioned in the June article that AOX had provided me with a program that allows you to change the operating speed on the Z-Master 386. As you may recall from the January article, the ZX-386 has speed changing capability built into its ROM, and you can change between full speed (16 or 20 MHz) and 6 MHz operation by pressing special key combinations. The ZX-386 appears to accomplish speed changes by switching between two oscillators. The Z-Master 386 board arrived with no indication in its manual that the speed could be changed, and I only found one processor oscillator on the board. So I figured that it was not possible to change its speed. However, after I had sent AOX a copy of my June article, in which I originally mentioned that the board had no speed change capability, I was informed that it did have that capability, and that AOX would send me a disk containing speed changing software. Unfortunately, the disk arrived the day after the June issue went to print, so I was not able to report on the software.

As it turned out, the AOX disk contains two speed change programs. One is a device driver that allows you to change speeds by pressing certain key combinations (as with the ZX-386). The other program allows you to change speeds at the command prompt, and it also allows you to set how slow the slow speed will be. You can set the slow speed to several different settings, ranging from 3 MHz to nearly as fast as full speed. I have no idea how the speed changes are accomplished, but it works. Games that run too fast at full speed seem to work just fine at one of the slower speeds, and that, to me, is the acid test of a speed changing capability.

The Mystery of the Cartridge Drive

I have a Syquest removable cartridge hard disk drive in the computer that I used to evaluate the ZX-386 and Z-Master 386 boards. I only use the drive occasionally, for backups, since the computer takes much longer to boot up when the drive is enabled (the drive takes a long time to become "ready"). But while I had the ZX-386 installed, with the old ROM in it, I decided to try the disk interleave test (one of the built-in test utilities) on a cartridge disk. I got a "read error" message before the test was completed. I ran the test on other cartridges and got error messages on nearly every one. I decided that because the cartridges were used ones (the kind we sell on the HUG Bulletin Board "Bargain Centre"), that there was no other problem that could be causing the errors. However, I later observed that when I tried to read data from a known good cartridge, that the drive would occasionally appear to "seek" more than it usually did. I ran my DTEST disk testing program (from HUG disk 885-3052) on the drive, and although it did not find any bad sectors that were not already recorded in the File Allocation Table (FAT), it did halt its normal progress of stepping though the disk's sectors several times. It appeared to be having trouble reading parts of the disk. On a hunch, I slowed the ZX-386 down to 6 MHz and tried the test again, and this time there were no halts in the test, except where there were actual bad sectors.

I removed the ZX-386 and replaced it with the Z-Master 386, and found that the Syquest drive could read its cartridges with the board at full speed without problems. I called AMI and told them about the problem with the Syquest drive, and they suggested that I run the Extended Setup and change the bus speed from 1/2 the processor clock (or 10 MHz) to 1/3 the processor clock (or 6.67 MHz). However, when I ran Setup, I found that the bus speed was already set to 1/3 the processor clock, and could not be made slower. So it appears that, for some reason, the ZX-386 cannot reliably access a Syquest drive (or at least my Syquest drive) while it is running 20 MHz. I do not know if the 16 MHz version would have such problems, and I should point out that I have had absolutely no problems with my fixed hard disk with either the ZX-386 or the Z-Master 386 installed.

One Port or Two

Although I pointed out in my January article that the ZX-386 provides two serial ports, I failed to point out in my June article that the Z-Master 386, which uses the existing Zenith I/O card that contains one serial port, does not provide any additional ports. If you need two serial ports, the AMI upgrade will provide them and save you the slot that you would have to use for a serial port card with the AOX upgrade.

Since my June article, I have finally gotten around to trying out some software that I have heard much about. It is QEMM-386, or the Quarterdeck Expanded Memory Manager for 80386 systems, from Quarterdeck Office Systems. It is available by itself, or as part of Desqview 386. For those who need to run more than one MS-DOS program at a time, Desqview 386 is about the best thing to have. But for any 80386 computer owner who wants to get the best performance out of his machine, QEMM itself is one of the cleverest things since sliced bread. With it, you can do any or all of the following things.

- You can convert some or all of your extended memory (memory beyond the first megabyte) to EMS memory. EMS memory is memory beyond the first megabyte accessed in a way so that certain DOS programs, such as Lotus 1-2-3, can use it. Normally, you need special hardware to have EMS memory. This can be either an EMS memory board, or, as in the case of the Zenith Z-386, a switch on the memory board that lets you select whether it is to be regular extended memory or EMS memory. With the Zenith memory board switch, you cannot make half of the board EMS and half of it extended, but with QEMM you can.
- You can map some of your extended memory into unused space between the "DOS limit" at 640k and the top of the first megabyte. The reason why DOS is limited to 640k is not because a DOS program cannot access memory above that limit, but because the memory above that limit has been reserved for video cards, ROMs, etc. But usually only a small part of that "space" is actually taken up by video cards and ROMs, and the rest goes to waste. QEMM takes advantage of the 80386 processor's ability to assign a virtual address to a block of memory that is different from the physical address. Therefore, it can take some of the memory above 1 megabyte and make it appear to your DOS programs that it is in the empty spaces between 640k and 1 megabyte. You can use this memory for running your TSR (Terminate and Stay Resident) programs, such as HEPCAT (HUG part no. 885-3045), device drivers (such as ANSI.SYS), and you can also use it for extra DOS buffers (which speeds disk accesses). You can have your favorite TSRs in memory and still have plenty of space for running application programs.
- You can map some of your extended memory into the space occupied by your system ROM and your video card ROM, and "shadow" the code from the ROMs into that memory. Because ROMs are slow devices, anytime your computer must run a routine that is in a ROM, it has to slow down. But if the ROM code can be copied into faster RAM memory at the same address, it can run at full speed.

The more recent models of Zenith computers accomplish this automatically in hardware, which is one of the reasons why the 16 MHz Z-386 was able to keep up with the 20 MHz AMI and AOX boards in my January and June articles. QEMM can do ROM shadowing without special hardware, adding its advantage to just about any 80386 system.

I first tried QEMM-386 on my H-386 system at work. After finding that I had to change some of the defaults to make it work (for example, I had to set the "Page Frame" address to D000-DFFF), I soon had it working properly, and now CHKDSK shows 603,680 bytes of free memory for applications even though I have 4 TSR programs loaded (one small one in conventional memory), 2 device drivers, and 40 disk buffers. One of the device drivers is VDISK, set up to provide a 1 megabyte memory disk, and I have nearly 2 megabytes of EMS memory available. All of this memory is derived from one 4 megabyte Zenith memory card.

After this success, I was considerably disappointed when I tried QEMM-386 on my two upgrade boards and found that, at first, it would not work on either of them. The computer would boot up to the point where the QEMM.SYS device driver was loaded (the heart of QEMM-386) and then crash. The ZX-386 would start the boot process over at this point, while the Z-Master-386 would just freeze up. The manufacturers of both boards said that QEMM-386 worked fine with their boards, so I figured that it must be something else in my system. I was able to find the problem by running QEMM.SYS with all of its features disabled. Then I ran the companion program QEMM.COM, which indicates what QEMM.SYS "thinks" about your memory above 640k. It indicated that some of the memory used by my Z-549 video card's ROM was "Rammable". I found that I could overcome the problem by using an EXCLUDE statement in the CONFIG.SYS command line for QEMM .SYS. Later, I found that I could instead use a ROM statement with the address of the video card ROM specified, which causes QEMM to shadow the video card's ROM.

There was an additional problem with the ZX-386 board in that QEMM.SYS would not run as long as the board's own ROM shadowing feature was enabled. So I disabled the ROM shadowing using the Extended Setup and let QEMM do the shadowing instead. Therefore, if you are thinking of getting a ZX-386 and QEMM-386, you might want to specify the ROM with the built-in test utilities rather than the ROM with the Extended Setup, since you cannot use the main feature (in my opinion) of the extended setup, which is ROM shadowing.

For those of you who are planning to try QEMM-386, the CONFIG.SYS command line that will run QEMM.SYS with all of its features disabled.

```
DEVICE=QEMM.SYS FRAME=NONE
```

The command line that works on both upgrade boards in my system is:

```
DEVICE=QEMM.SYS ROM=C000-C7FF
    ROM=F000-FFFF RAM
```

This command line enables ROM shadowing of both the video and system ROMs (assuming a Z-549 video card). It also maps extended memory into the unused spaces between 640k and 1 megabyte. The command line I actually use on my system is:

```
DEVICE=QEMM.SYS EXCLUDE=B000-B7FF
    ROM=C000-C7FF ROM=F000-FFFF RAM
```

This command line causes QEMM .SYS not to use the memory from paragraph B000 to B7FF, which is the memory normally used for monochrome video. I have a color monitor on my system, so you may be wondering why I would exclude that section of memory. It is because with the Z-549 video card (and also with the Z-449 and HVB-550 video cards) and an analog monitor, you can make the system emulate a Hercules monochrome video system, and run programs that work best in monochrome. I occasionally use the program NewsMaster II, which looks much better when it is run on a Hercules monochrome graphic system than on a color graphic system.

If you are wondering how to find out where your ROMs, video boards, and unused spaces are in the memory space above 640k, there is a utility program called System Sleuth available from DTG of El Toro, California that can provide that information. If you run System Sleuth after QEMM has been installed, it may provide erroneous information about your high memory, but if you run it without QEMM, it usually provides accurate data. Then you can install QEMM using the data provided. System Sleuth can also provide information about your I/O ports, interrupts, installed device drivers, and disk drives. It costs a bit much for something that you would not use often, but it certainly is handy when you need it.

Product Information

The ZX-386 is a product of American Micronics, Inc., 18005 Skypark Circle, Irvine, CA 92714, (714) 261-0780. One of their distributors, SMS Data Products Group, sells some AMI products at special prices to the government, and they have decided to offer those products to HUG members at the same special prices. Here are the models that are available from SMS.

Item	HUG Member Price
1 MB 20 MHz ZX-386	$1945
2 MB 20 MHz ZX-386	$2495
4 MB 20 MHz ZX-386	$3595
8 MB 20 MHz ZX-386	$5795

These special prices are available from SMS Data Products Group, Inc., 1501 Farm Credit Drive, McLean, VA 22102, (703) 827-0640.

The Z-Master 386 is a product of AOX Incorporated, 486 Totten Pond Road, Waltham, MA 02154, (617) 890-4402. Certain AOX distributors provide a 20% discount to HUG members. To get the 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	Discount Price
16 MHz Z-Master 386	$1196
20 MHz Z-Master 386	$1400
25 MHz Z-Master 386	$1756

To help you compare the prices of the two upgrades, consider that the SIMM memory modules required to populate the Z-Master 386 cost about $75 to $100 each for the 256k size, and about $200 to $300 for the one megabyte size. You will need four modules to populate your Z-Master 386. You can get your memory modules from one of the more reputable mail order companies, such as those that advertise in REMark. Memory prices are unstable, but these figures should give you a rough idea of how much to add to the Z-Master price (as of this writing).

QEMM-386 and Desqview 386 (which includes QEMM-386) are available from Quarterdeck Office Systems, 150 Pico Blvd., Santa Monica, CA 90405, (213) 392-9851. QEMM-386 is $59.95, and Desqview 386 is $189.

System Sleuth is available for $149 from DTG, 23704-5 El Toro Road, Suite 348, El Toro, CA 92630, (213) 987-2000. ✱

Continued from Page 6

the expanded memory to drive E.

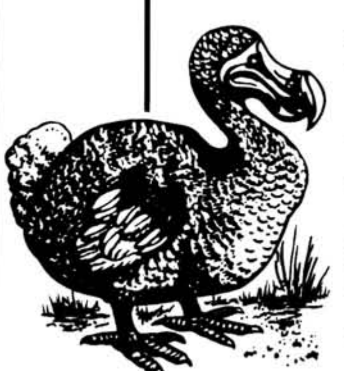
Things seemed to be working properly until I rebooted the system. Now I couldn't find either drive D or E, although I could find D after rerunning "Asgnpart". I finally found my expanded memory on drive G. I only mention these problems as the solution may be of help to someone.

As stated in the article, it seems that when the Automatic Position Assignment Flag is set to automatic only one partition (C) is recognized, but when set to manual, four partiitions can be recognized. It should be noted that in the manual mode it reserves four partitions (C-D-E-F) no matter how many of them are assigned. This explains why the expanded memory was on G. I also discovered that the assignment of drives D-E-F does not survive a reboot and must be done after each one. I added this to my Autoexec file and also changed Lastdrive in config.sys to G. Now everything is working fine and I have the system that I had almost given up on.

Thanks for the help that your article gave me. I find many of the articles in REMark interesting and useful. Keep them coming!

Sincerely,
Robert W. Tripp
1055 Peppertree Drive
Sarasota, FL 34242

✱



HUG

Discount List

AA-3000	20%
AB-3000	20%
CB-5063-50	10%
DC-1000	10%
EN-2000	20%
HA-2860-1	10%
HA-2860-2	10%
HA-2860-4	10%
HBT-40-AT	10%
HBT-40-H4	10%
HBT-40-HZ	10%
HBT-40-SD	10%
HBT-40-XT	10%
HS-40A	10%
HS-42	10%
HS-386C	10%
HS-2526A	10%
HS-2860	10%
HSM-100	20%
HSM-100-3	20%
HV-2000	10%
HVB-550	10%
HWD-420	10%
HWD-440	10%
HWD-4028	10%
IN-6000	20%
PA-120	20%
PD-500	10%
PM-160	20%
PM-160-3	20%
PMK-121	10%
PMK-130	10%
SK-203	10%
SK-209	10%
SK-211	10%
TM-140	10%
TM-150	10%
TM-158	10%
TM-159	10%
TM-170	10%
TM-200	10%
TM-240	10%
TM-380	10%
Z-207-7	10%
Z-304	10%
Z-315	10%
Z-316-8	10%
Z-405A	10%
Z-416-2	10%
Z-416C	10%
Z-445	10%
Z-449	10%
Z-505	10%
Z-515	10%
Z-516	10%
Z-525	10%
Z-605-1	10%
ZA-180-35	10%
ZA-180-39	20%
ZA-180-40	10%
ZA-180-57	10%
ZA-180-62	10%
ZA-180-63	10%
ZA-180-65	10%
ZA-180-66	10%
ZA-180-67	10%
ZA-181-4	10%
ZA-181-7	10%
ZA-181-9	10%
ZA-181-17	10%
ZA-181-19	10%
ZA-181-24	10%
ZA-3034-AS	10%
ZA-3034-CS	10%
ZA-3034-EB	10%
ZA-3034-ME	10%
ZA-3034-NP	20%
ZB-315-1	10%
ZBV-2526-EK	20%
ZBV-2527-EK	20%
ZBV-3339-KQ	20%
ZBV-3340-KQ	20%
ZBO-2503-EK	20%
ZCA-2300-EF	10%
ZCA-2300-MG	10%
ZCF-2325-EY	20%
ZCM-1390Z	20%
ZCM-1400-1	10%
ZCM-1490Z	20%
ZD-12	10%
ZD-14	10%
ZD-72	10%
ZD-200	10%
ZD-400	10%
ZD-800	10%
ZDF-1211-DY	20%
ZDF-1217-DY	20%
ZDF-2237-BK	20%
ZDF-2255-BK	20%
ZDH-1211-DE	20%
ZDH-1217-DE	20%
ZKB-2	20%
ZMM-149A	20%
ZMM-149P	20%
ZMM-1470G	20%
ZSS-184-1	20%
ZSW-184-2	20%
ZTC-3034-EB	20%
ZTC-3034-MO	20%
ZUS-386	10%
ZVM-135	20%
ZVM-1200-1	10%
ZVM-1240	20%
ZVM-1300-1	10%
ZVM-1330	20%
ZVM-1380C	20%
ZWL-200-2	20%
ZWL-200-4	20%
All software included in the software chart shown on pages 88 and 89 of catalog #213	20%

On The Leading Edge

SuperCalc 5, Z-386/33, MIPS and Performance, SupersPort 286, Zenith Support

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

It happens to all of us sooner or later. You spend your hard-earned money on something for your computer, either hardware or software, and it doesn't perform as you expect. Or even worse, it doesn't work at all. That situation has got to be one of the most frustrating problems that you have to deal with. When something does not work like you expect, it may be something very small that results in an inordinate amount of frustration and dissatisfaction. Such is the case with a software upgrade that I recently bought.

SuperCalc 5

I have used SuperCalc in all versions since its release a number of years ago. My first experience with SuperCalc was using it with the CP/M operating system on my H-89 to do the budget calculations for my department. SuperCalc was around long before anyone ever thought of Lotus 1-2-3, and I have never seen a version of SuperCalc which was copy protected. Unfortunately, the company which owned SuperCalc (I think it was Sorcim at the time) did an incredibly poor job of marketing, and Lotus 1-2-3 became the "standard" for a spreadsheet on the PC compatible systems. I have always found that SuperCalc was faster and easier to use than Lotus, and in many cases, it was more powerful too. I've always thought that SuperCalc's menu structure was much more intuitive and more logical than Lotus 1-2-3, and SuperCalc generally requires fewer keystrokes to perform the same task. The Lotus' requirement of preceding formulas with a "sign" (e.g., +) and the required use of the "at-sign" (@) for all of the built-in functions is quite unnecessary. In short, I have always liked SuperCalc and have mentioned it in this column.

SuperCalc 5 has lots of new features, and Lotus version 3.0 may even have some of them too, if it ever gets released. I have no real complaints about the functionality of the new SuperCalc since everything seems to work pretty much as I expected. My complaint is almost trivial in terms of functionality, but is very real based on a comparison with other similar software.

SuperCalc 5 has no way to customize the colors for the screen display. You are stuck with the same old yellow column/row bars, white text (black background), and a blue cell indicator. I have also used Borland's Quattro spreadsheet since it was released last year, and although it may not have all of the features of the current SuperCalc release, you can change all of the colors to suit your own preferences. As I have mentioned before, I generally prefer black text (or numbers) on a white background because I think it is easier to read. That particular color combination also seems to help minimize glare problems on older CRTs, especially in well-lighted offices. And although it seems churlish to criticize a program with as much functionality and features as SuperCalc 5 because of this omission, it seems to me that SuperCalc has not progressed beyond 1984 in that respect. For my part, I have returned to using Quattro because I like the color display for a spreadsheet, even though I still have to use the Lotus "standard" of preceding built-in functions with the "at-sign." I have found that a color display makes my work so much easier that I can afford to overlook some other things.

The Z-386/33

According to the April 17, 1989 *InfoWorld* (page 27), Zenith began shipping the 33 MHz Z-386 in April. There are several different models ranging in price from $7,999 to 13,499, depending on the configuration. All models include a standard 2 MB of system memory, 16 KB of cache memory, a 3.5" 1.4 MB floppy drive, and an ESDI hard disk controller with a 1:1 interleave. The release also states that this system will deliver more than 8 MIPS of performance.

As we move toward faster and more powerful computers, you will begin to see that more and more of them talk about MIPS — Millions of Instructions Per Second. This measurement of computer performance has been used in mainframe systems for years, so it is worthwhile to take a minute to understand what this performance measurement really means.

Measuring Performance

MIPS is defined as the average number of machine language instructions that a computer can perform in one second. Because of the way MIPS is defined for an *average* number of instructions, it is frequently difficult to get people to agree on a MIPS measurement for a specific computer. The technical reason for that is simply that some machine language instructions (i.e., assembler) are "longer" than others — some instructions are one byte long, some are two, and a few are three bytes long. Those of you familiar with assembler on any computer (mainframe or micro) are no doubt familiar with that fact. Given that information, it is probably obvious that one can write a program with "short" instructions that will show a better value for the MIPS measurement, even though the "standard" says that the entire set of machine language instructions should be exercised. Still, a performance value in MIPS is still far better than comparing clock speed on a computer because, by its definition, MIPS takes into account such things as clock speed and memory wait states.

To give you some idea of the various performance values, I looked through some IBM documentation to find out some of the obvious MIPS values. The original IBM PC was rated at 1/4 MIP (250,000 Instructions Per Second), and the IBM AT was rated at 3/4 MIPS (750,000 Instructions Per Second). Since the Z-241 did not have any memory wait states, Zenith told me that it was rated at approximately 0.94 MIPS. If you assume that these performance values were all rated by the same program, then you can conclude that the new Z-386/33 is over 32 times faster than the original IBM PC.

It wasn't all that long ago that mainframe computers were less powerful than this new Zenith 80386 system that operates at 33 MHz. For example, the old IBM 3033 mainframe system ran at 2.4 to 8.0 MIPS, depending on the configuration, but that was not the end of the story. The Z-386/33 certainly weighs less than 100 pounds while the IBM 3033 weighs in at several THOUSAND. In addition, you need special power, heavy-duty air conditioning, and chilled water support for this mainframe that is obviously not required for a desktop computer. By the time you finished installing a 3033 system, it was easy to spend several million dollars on the hardware and support equipment. Now you can buy the equivalent computing power for under $15,000 in a desktop system — a truly remarkable testimonial to the hardware advances that have been made in the last 15 years.

Given that the Z-386/33 provides about 8 MIPS of computing power, where does that place it in terms of today's mainframe computers? If you have an extra few million dollars, you can buy an IBM 3090 series computer that will give you up to about 75 MIPS (Model 600E) of computing power, which is a little more than I really need for word processing. Or, if you have an extra 20 million dollars or so, you can buy a Cray supercomputer that provides over 100 MIPS of performance. I hope that this information gives you a perspective of where all of this power fits into the overall scheme of data processing, and I think it explains why these desktop systems have been so popular. For cost versus performance considerations, desktop systems are impossible to beat.

The SupersPort 286

As I have mentioned from time to time, I also do a considerable amount of computer consulting, mostly for mainframe-based systems. One side effect of this kind of consulting is that I also have an opportunity to work with a wide variety of microcomputer hardware and lots of different software, such as word processors and graphics packages for reports and presentations. My reports to you on software are based on this experience with a large number of similar packages which I hope gives you some ideas of what may work best for you. Working with lots of different software has its disadvantages though. Perhaps the biggest disadvantage is the time required for the learning curve for software that I am not familiar with. Like most people, I can work much faster if I use software that I know, and that was the primary reason that I bought a Z-171. Although the Z-171 is an excellent computer, it became apparent that I needed a hard disk system when I was traveling. I did not need a hard disk so much for speed as I did for its storage capacity. For example, I use the GEM Presentation Team software a lot, and that takes a couple of megabytes of disk space. Of course, I use word processors a lot, and even WordStar requires a lot of disk space, not to mention Microsoft Word and some others. Carrying a lot of floppy disks was becoming a real chore, so I finally decided to take the plunge and bought a SupersPort 286. As a side note, this is the first Zenith computer I have bought that is also available as a Heathkit, although I normally buy the kits because I like to assemble them myself. In this case, I was rushed for time, and I really did not have the few hours required to assemble the kit version.

Since I needed the hard disk capacity in the first place, I decided to get the SupersPort 286 with a 40 MB hard disk. I have found that it is quite easy to fill up a 20 MB hard disk on my desktop systems, and I did not want to get stuck with a hard disk that quickly became too small in a laptop. I also chose the SupersPort 286 because it seemed to provide the most "bang for the buck" for my purposes. Although the TurbosPort 386 has a nicer display that is EGA compatible, I thought it was a bit too pricey for my needs. And there is a trend that the newer software will require at least an 80286 CPU, so I decided that my choice would have to have at least that.

I spent the day (Saturday) after I got the new laptop fooling around with it and loading software. I fired up the PREP command for the hard drive, and that took just under an hour to complete. I then loaded Zenith MS-DOS 3.3 Plus followed by the tedious installation of a lot of application software — various word processors, spreadsheets, and the GEM Presentation Team. The following day I left for a consulting assignment in New York City.

The SupersPort performed perfectly during the first week, and I used all of the software that I normally use on my desktop systems. Hindsight makes it much easier to see how much time I would have saved with a hard disk system, and if you are considering a laptop, I strongly recommend that you get one with a large hard disk.

Monday morning of the second week of the assignment in New York started out in the usual chaotic way which always seems to happen when traveling. I was working on several things that morning using the Samna IV word processor, and I powered-off the system to go to a meeting. When I returned and powered-on the system, it would not boot from the hard disk. Of course, I had to finish something for a meeting that afternoon, and Mr. Murphy decided he would "fix" my system at the worst possible time. Instead of executing the usual panic routine, I took my own advice at this point and decided to take a coffee break to help relax and think through the problem. Since I had been using the system with no problems up to the point of powering it off, I decided that the hard disk must generally be okay, especially on a new system, and that was easy to check out since I had a bootable floppy with me.

Fortunately, the hard drive was completely accessible when I booted from a floppy, and I ran CHKDSK on both partitions to see if there were any obvious problems. There were not. The directories and File Allocation Tables on both partitions were apparently intact, so I checked out some of the applications, and they ran fine too. Since I was able to get to all of the programs and files on the hard drive, I finished my work in time for the meeting, even though I had to skip lunch to do it.

I finally had time for reflection on the technical problem later that afternoon, and it was obvious that at least one system file somehow got clobbered. What was particularly puzzling about the problem was, when the system attempted to boot, it apparently stopped. No error message, no nothing. The hard drive light came on for an instant, but it did not flicker like it normally did during the boot process. Since it was quite apparent that I had some kind of a problem with the system files, I followed a reasonable procedure. First, I used the SYS command to transfer new copies of the BIOS (IBMBIO .COM) and the System Kernel (IBMDOS .COM) to the hard drive. Then I used the COPY command to copy the Command Interpreter, COMMAND.COM, to the C partition. That should fix everything, right? Wrong! I still had exactly the same problem, and the hard drive would not boot. The system was still not kind enough to display something like a "non-system disk." At this point I had a strong suspicion as to what the problem was, but I could not believe it, so I started calling a few people for some suggestions.

Because I was traveling over 1,000 miles from home, I did not have the luxury of recovering the hard drive in the way I normally would. I finally talked to a friend of mine who suggested that about the only way that the symptoms I described could occur was if something in the boot sector was clobbered. If something in the loader code gets messed up, it is possible that the system will appear to hang and no error message will be displayed because the boot process does not get that far.

I used DEBUG to look at the boot sector, and sure enough, there were all kinds of strange things that appeared where the boot loader should be. I saw things like COM1, COM2, CLOCK$, LPT1, and LPT2 that simply are not part of the code in the boot sector. I even retyped the DEBUG Load (L) subcommand a couple of times to be sure that I had not made an error when I loaded sector zero to examine the code. As it turns out, I learned that stuff is part of the BIOS file (IBMBIO.COM), and it is not even remotely related to anything that should be in the boot sector. I am indebted to my friend Brian for helping me diagnose the exact nature of the problem. Now I knew that the boot sector was bad, and there were several alternatives.

The easiest by far is to simply back up the partition, and rerun the FORMAT command to recreate the boot sector. Unfortunately, I did not have that option because I did not have enough floppies with me to back up the partition nor did I have any backup copies of my MS-DOS masters. Even though that sounds like a clumsy excuse because it really is, it was simply not possible to do that because of the circumstances. I had set up the SupersPort system so quickly that I did not take a lot of the normal precautions, such as backups, that I normally do. Since I really wanted to be able to boot the system from the hard disk, there was only one alternative left — recreate the boot sector by hand with DEBUG. I won't go into all of the gory details of how I did that, but suffice it to say that I can think of better ways of spending an evening in New York. In any case, I was able to get the system up and running so that the hard drive would boot successfully.

During this time, I kept wondering what caused the boot sector to get clobbered in the first place. If you have any technical knowledge of DOS and how it works, you will understand why the boot sector is normally written only by the FORMAT command, and although other programs may read certain information in the boot sector, such as disk format information, it is generally forbidden to write anything to the boot sector. If you know how, you can use something like DEBUG or HADES to write to any logical sector, but that is the exception to the rule. With those exceptions, the boot sector is generally off-limits to application programs.

Because I wanted to avoid clobbering the boot sector again, I tried to recall exactly what I had done that morning that might have caused the problem. I had been using Samna IV for word processing all morning, and I had used it the previous week with no problem. So, I must have done something unusual with Samna. Aha! I remembered that I had shelled out to DOS from Samna to execute a command from the command line, and when I returned to Samna (with the EXIT command), strange things began happening. I could not remember exactly what command I had used, but it was probably a DIR command to check the files in a subdirectory or on a floppy. After returning to Samna, I saved the current file, powered off the system, and went to a meeting as I mentioned earlier. For those of you familiar with Samna, I used the F9 (DO) followed by an exclamation point (!) to shell out to DOS. After some thought, I realized that was the last thing I did, and it was the only unusual command I had executed during the morning. All of the other commands were related to formatting in one way or another, such as bolding, underlining, etc.

There is evidently a bug somewhere in this particular Samna IV function, so the simple, and easy way, to avoid the problem is to avoid using that particular command. That was no particular surprise to me because I regularly had to recover files that were clobbered for one reason or another in Samna III, but I did find it unique that this version of Samna somehow managed to clobber the boot sector. As software becomes more complex, I guess it is inevitable that these kinds of bugs will be inadvertently introduced in the programming, but this particular bug is especially bad. Although it turned out that I really did not lose any data as a result, the experience was certainly disconcerting. Even if I made some kind of user error that I can't really be sure of, I don't think any software should cause an error like this under those circumstances. It is unfortunate that computer users have to cope with these kinds of problems, but I doubt that Samna is unique with respect to serious bugs.

Even with that problem, I have been quite pleased with the performance of the SupersPort 286, and it was certainly an excellent choice for my applications. So far as this computer is concerned, I really have found only one problem with it that is not unique to laptops in general. The cursor is sometimes very difficult to see, especially when it's at the bottom of the display. Some software, like WordStar 5, allows you to change the shape of the cursor to a block form which is much easier to see, but that is not always the case. I even tried Pat Swayne's NOBLINK program, but it did not work with Samna because that program always reset the cursor to the nearly invisible underline. It would be a big help if all software manufacturers realized that these programs will be used on a wide variety of display types and provided configuration options which would make software easier to use.

FORMAT with MS-DOS 3.3 Plus

As I have been working with the SupersPort 286 and my Z-386, I noticed some real changes in the FORMAT command for MS-DOS 3.3 Plus. Both computers have one common feature — a 1.44 MB 3.5" floppy drive — and I have observed that many people are having difficulty formatting a 720 KB disk in that drive with this new DOS version. If you don't know exactly how to format a disk in a drive, the Zenith manual is not very helpful in what to do. And the error message displayed is something like "Track 0 bad — disk unusable" which is no help at all in solving the problem. Although there are several ways to solve the problem, I suggest the easy way, but there is one important fact that you must know which is not in the manual.

The FORMAT command always attempts to format a disk at the maximum density of which the drive is capable unless you tell it differently. If you have a 5.25" high-density drive (1.2 MB) or 3.5" high-density drive (1.4 MB), FORMAT will attempt to initialize the disk at 1.2 MB or 1.4 MB respectively. If you don't use a disk that is certified for that capacity, you will usually see an error message. For example, a double-sided/double-density 3.5" disk is certified for 9 sectors per track, and 3.5" high-density disk formats at 18 sectors per track. I suppose you could say that the high-density disk is a "higher quality" disk because it can store twice as much data per track as a double-density disk. You will also find that these higher-quality high-density disks are considerably more expensive than the standard double-density disks. Still, it is important to know that the FORMAT command attempts to format based on the DRIVE type, not the disk type.

My Z-386 has a 1.2 MB 5.25" floppy drive (drive A) and a 1.44 MB 3.5" floppy drive (drive B). The SupersPort 286 also has a 1.44 MB 3.5" floppy drive as drive A. Although you can fool around with the ROM-based SETUP command and/or the Zenith DSKSETUP command to change the configuration, I suggest that the physical drive types be defined exactly as what they are. Then you can use the FORMAT command parameters to define precisely how you want the disk formatted.

Most of these commands will work with Zenith MS-DOS versions prior to 3.3 Plus with the exception of the /N parameter which is new to this version. The first general form of the FORMAT command is the usual one as shown in Figure 1.

```
FORMAT d:[/S]
```

Figure 1
FORMAT Drive d to the Maximum Drive Density

Whenever you use the FORMAT command, I recommend specifying the drive letter *d* to help prevent formatting the default drive. The /S option can be used if you want to create a bootable disk. Keep in mind that this version of the command will attempt to FORMAT a disk at the maximum density available for that drive. It does not matter what size or type of drive you are using, this command will still try to format at the maximum density. For 5.25" drives, that means the command will format at 360 KB (double-density) or 1.2 MB (high-density), depending on the drive type. For 3.5" drives, the command will format at 720 KB (double-density) or 1.4 MB (high-density), depending on the drive type.

So much for the defaults. But what if you want to format a 360 KB disk in a 1.2 MB drive or a 720 KB disk in a 1.4 MB drive? That gets a little trickier. In order to avoid problems, let me repeat that you MUST have your system configuration correctly set to the type of PHYSICAL drive. Yes, there are a few ways around this using the ROM SETUP command and/or the Zenith DSKSETUP command, but I have not tried all of the possible combinations and permutations of those commands. I have tried the correct setup on my Z-386 and SupersPort 286, and the following commands always work correctly when the system configuration is set properly. Assuming that your system is correctly configured, here's how you can FORMAT a double-density disk in a high-density drive.

```
FORMAT d:/4[/S]
    (360K disk in 1.2 MB drive - 5.25")
FORMAT d:/N:9[/S]
    (720K disk in 1.4 MB drive - 3.50")
```

Figure 2
FORMAT Double-Sided Disk in HIgh-Density Drive

As in the previous example, I recommend specifying the drive letter *d* to help prevent formatting the default drive, and the /S option can be used if you want to create a bootable disk. For a 5.25" disk, it is easy to remember that you must use the /4 switch to format a double-sided disk in a high-density drive because 360K is about one-fourth of 1.2 MB. The /4 switch is available in virtually all version 3 release of DOS.

To FORMAT a double-density 3.5" disk in a high-density drive, version 3.3 Plus has a "new" switch /N, which is the number of sectors per track on a 3.5" disk only. Specifically, you must enter the complete switch value of /N:9 which tells FORMAT to initialize at 9 sectors per track for a 720K (double-density) disk in a high-density drive. Because FORMAT looks at the drive type, it will attempt to FORMAT at 18 sectors per track (default) on a high-density drive when this switch is not specified.

Although the /N switch has a new meaning in version 3.3 Plus, it is not really a new switch parameter for Zenith MS-DOS users. In prior versions, the /N meant "No prompt" so that you could use FORMAT in a batch file without having to enter any responses like "Press RETURN to FORMAT drive d." If you used the /N switch in a batch file in prior MS-DOS versions, you will have to change it to /Q (Quiet) to avoid an error message. FORMAT will not accept the /N switch without a number in this new release.

As in the prior release (version 3.21), the /V switch will prompt for a Volume label, and the /S (System) switch is used to create a bootable disk as I have mentioned. This version also includes another new switch /T:nn which specifies the number of Tracks on a 3.5" disk, but I have not found any real need for it because the default is 80 (i.e., /T:80), and there is no need to use it. Many of the other switch parameters are the same as previous releases, and you should check the manual if you have any other special formatting requirements.

Zenith Product Support

Despite some apparent attempts to improve product support, it is quite evident that Zenith has done a really poor job in this area. In the last year or so, I have noticed consistently poor reports of virtually all Zenith products, and the latest one appeared in the April 17, 1989 *PC Week* (page 108). This magazine has conducted a series of articles which are called "The PC Week Poll of Corporate Satisfaction," and this particular article was a poll of 286-based systems available through mail order. The Zenith Z-248/12 came in dead last in this poll, both in overall score and most notably in product support (68). Although this is the first time that I can recall Zenith coming in last in the overall ratings, Zenith has consistently performed poorly in the product support area. In contrast, the Dell System 200 received the highest score in this group with a very high score in product support (84). Based on the letters I have received from you, and some other checking that I have done, there are several possible reasons for this.

To place this situation into its proper perspective, it is important to remember two key factors in this poll. First, it is based on computers available through mail order which of course means that there is no local dealer to provide any support. And second, this is a poll of corporate users which means one can reasonably assume that there is at least one or more people on a support staff that understand the technical requirements of PC compatible computers. Given that scenario, I think it is especially interesting that Zenith is consistently getting a poor rating in product support. This becomes important because it is a *market perception* not just an opinion from a few individual users. What is Zenith doing wrong?

A little reflection of the past few years of sales makes it clear that senior management at Zenith Data Systems made a serious strategic blunder in concentrating mostly on the government sales and pretty much ignoring "small" users. In this context, I am using the word "small" to include any single user or group of users that did not have multi-million dollar contracts with ZDS. At the 1987 HUGCON in Chicago, ZDS president John Frank made some comments to the effect that "ZDS was really not interested in small users." I don't remember any exact words, but I vividly remember sitting at a table with other long-time, die-hard Huggies (like me), and we all looked at each other in amazement when he made these comments to an audience that was quite "friendly" and interested in his comments. At this point, it does not really matter what Frank said, but I also remember hearing more than one person say afterwards: "If that's his idea of the market, I will NEVER buy a Zenith computer." After Frank got finished with his comments, it was obvious that both Heath president Bill Johnson and Veritechnology president Joe Schulte recognized that Frank had carefully inserted his foot in his mouth clear up to the neck. Both Bill and Joe valiantly tried to "rephrase" Frank's comments, but it was clear that Frank had already made a major blunder in his first three weeks as ZDS president.

In the past, ZDS has apparently placed such importance in these large contracts that they have ignored just about everyone else. The major contracts have just about disappeared, and Zenith is now finding the microcomputer market has changed to a great extent, and they are not able to cope with it very well. Corporate mail-order customers and individual users are obviously not favorably impressed with the current quality of support available from ZDS.

My research into this problem indicates that ZDS is furiously trying to correct it, but it will take some considerable time and effort to overcome the market perception. I believe that ZDS will be able to correct the situation in time, but the lack of planning and foresight for days without large contracts will continue to haunt their sales and product reviews for some time yet. I also have evidence that ZDS also managed to alienate some good computer dealers, and it will also take time to patch up those relationships if they are inclined to do so. I hope they are. One of the cornerstones of good product support is easy user access to good dealer support.

As I tried to take a holistic view of the product support problem, it was also evident that some of it may be caused by the deteriorating quality of the Zenith documentation, specifically the MS-DOS manuals. If you compare the manuals for the Zenith MS-DOS version 3.1 to the current 3.3 Plus documentation, you will find there is considerably less detail in the current manuals. I think that ZDS also underestimated the importance of these manuals, but I have noted some minor improvements to the 3.3 Plus manuals as compared to version 3.21. Lack of some general information about these systems can cause a problem, and perhaps Zenith should consider including a basic list of suggested third-party hardware that works in each system. For example, Zenith apparently makes a serial card to add a COM2 port to the Z-200 series (I haven't seen one), but what other cards will also work? Unfortunately, the answers to some questions are not obvious, and many require considerable technical knowledge to understand what is going on and why. That particular subject is especially confounding to new users who add a board to their system only to find out that it apparently does not work. Some blame Zenith hardware when that is really not the problem. I have been working on an article called "IRQ Mysteries and Hardware Conflicts" that should help users understand what is going on and solve problems related to adding a new board in the system. This article will explain what you must know about the subject when you add a new board to your system, such as a bus mouse or an additional serial port.

Powering Down

This year promises to be an interesting one for microcomputers. It appears that Intel will really try to push the 80486 chip, but I wonder if computers using that chip will be too expensive for most users. There are also some indications that OS/2 may be making inroads into the operating system market, but I believe that its success, or lack of it, will depend on its overall cost effectiveness. At this point, I have not seen anything that would seem to make it worth the considerable expense to convert to OS/2. Perhaps the software vendors will have some pleasant surprises for us with new software. The bad news is that I also expect that OS/2-specific software will be more expensive than similar DOS-based programs.

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

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

Products Discussed

MS-DOS 3.3 Plus (OS-51-3)	
List Price	$149.00
Mail Order with Update Card Only	49.00
SupersPort 286 w/20 MB HD	4999.00

Heath/Zenith Computer Centers
Heath Company Parts Department
Hilltop Road
St. Joseph, MI 49085
(800) 253-7057
(Heath Catalog orders only)

*

Glitches

In the June 1989 issue of REMark, the article entitled "Home Financial Management" on page 27, has the price of HFS-III Home Finance System by Jay Gold Software as $99.00. This price has been reduced to $49.95.

Also in the June issue, the article entitled "Getting Started With . . . Microsoft Word" is not by Ralph E. Camp, but was written by Jack W. Bazhaw.

BUG ZAPPING

Bug: I'd like to install 3.5" drives in my Z-159 (Z-157 or Z-158), but can't figure out what to buy. The cabling is also different. Help!

Zapped: A ZCA-16 cable package is available for installing the ZD-72 3.5" Floppy Disk Drive in a Z-157, Z-158, or Z-159 computer. The ZCA-16 cable package includes:

- #134-1884 4-conductor power cable for 3.5" FDD.
- #134-2016 34-conductor drive-to-controller card cable with two 3.5" FDD connectors and a 5.25" FDD connector.
- #134-194 5-3/4" × 1/2" × 1/4" adhesive foam tape (rubber gasket).

Instructions for installing the cable assemblies are contained in the ZD-72 3.5" FDD Installation and Operation Guide (#595-4093). The ZCA-16 package is available through Heath/Zenith Computers and Electronics Centers.

Bug: 20mb Winchester drive is inoperative in the Z-2200 series computer.

Zapped: Check for a short between the connector of the #134-1869, 34-conductor cable, and the drive housing at the Winchester drive. To correct, install a piece of insulator paper between the connector and the drive housing. A new 34-conductor cable (#134-2011) will be used in the future to eliminate this problem.

Bug: I'm having strange problems with the serial port on my Z-386 computer. Any suggestions?

Zapped: A problem was found in the WD16C450 UART (443-1543). To eliminate it as the culprit, replace it (U312 on the #181-6875-2H I/O board) with an NS16450 (443-1281).

Bug: I upgraded my H/Z-248 (H/Z-241) to a '386 using the available upgrade kit. Now I have a number of modules left over which could almost make up another computer. What parts would I have to order to make up a minimum system with these leftover modules?

Zapped: After installing the ZUS-386 or HUG-386 upgrade kit, certain components will be required to rebuild the H/Z-241/248 computer. Not including the mounting hardware or video board, the list is as follows:

Heath P/N	Description
90-1368-1	Cover
94-654	Card guide adapter
94-682	Card guide
100-1860-4	Keyboard
181-7597	Lock assembly
200-1500	Disk drive chassis (2-required)
200-1561-1	Main chassis
203-2268	Front Panel
234-859	Power supply

The total cost for these parts will be between $650 and $750 depending upon current prices. Contact the Heath Parts Department for more information at (616) 982-3571.

Bug: The ZVM-1380 won't power up with the RGB cable unplugged or in EGA mode.

Zapped: When an RGB cable is not connected to the monitor, it automatically switches to the EGA mode. If the monitor fails to power up when the RGB cable is unplugged or you switch to the EGA mode, try changing anode voltage regulator transistor Q408 (#969-537). This transistor may be located on the main board or on the large aluminum heat sink.

Bug: My ZVM-1380 has poor display and the pincushion control has no effect.

Zapped: On the main board, check the power pincushion regulator transistor at Q405 (#969-522). Changing this transistor usually corrects pin amp and width trim control problems.

Bug: The ZCM-1390 monitor shuts down when the screen goes from near black to full white.

Zapped: Some ZCM-1390 monitors will shut down when the brightness and contrast controls are set at maximum light output and the user runs software that causes the screen to go from near black to full white. On the deflection board (Zenith P/N 9-717), make the following changes:

Change R3213 and R3214 from 5.6 kilohms and 1 kilohm resistors to 10 kilohms 1/4 watt, 5% film resistors.

This modification was implemented in units whose serial number ends in "TOD".

Bug: I went through the MFM-200 Monitor SETUP program and changed my video configuration to monochrome on my Z-200 computer. Now, when I boot my system, I cannot get any display on my screen. What did I do wrong?

Zapped: You probably chose the wrong video configuration in the SETUP program for the system that you have. Take the cover off your computer and completely remove the I/O board for a few seconds. This will erase all the information about your system stored by the SETUP program. Removing the battery will also accomplish the same thing. Replace the I/O board and cover. Check the documentation for your video board and monitor for configuration information before running SETUP again.

Bug: How do I get background and foreground colors on my screen?

Zapped: First, you must have CGA or EGA video card. Second, you must have color or enhanced graphics monitor.

At the operating system level, you can use the ANSI escape sequence characters for color. These codes can be found in your MS-DOS manual, Chapter 11 under ANSI.SYS terminal driver (for MS-DOS v2.1 and 3.1). They can be found in Appendix C for MS-DOS 3.2. To use the ANSI escape sequence, you must first have your ANSI.SYS device driver installed (Please refer to MS-DOS manual).

Continued on Page 44

#6 Z-100 SURVIVAL KIT

Paul F. Herman
3620 Amazon Drive
New Port Richey, FL 34655

Those Odd Escape Sequences

All of you are pretty familiar with the various escape sequences that can be used to control the Z-100's screen and cursor. (See Appendix B, "Symbols and Codes", of the Z-100 User's Manual.) For instance, 'ESC E' can be sent to the console in order to clear the screen. But what about those funny escape sequences that aren't so self-explanatory? The ones that no one ever explained how to use? Well, it's about time we learned what they do, and how to use them.

The escape sequences I'm talking about all transmit information from the console, back to the computer. One of them is used to obtain the cursor position;

```
ESC n          Cursor position report
```

Two of them are used to identify the terminal type;

```
ESC Z     Identify as VT52
ESC i 0   Zenith identify terminal type
```

And four of them are used to transmit characters from the console;

```
ESC _     Transmit character at cursor
ESC ^     Transmit current line
ESC ]     Transmit 25th line
ESC #     Transmit page
```

To do justice to this thing, I'll have to split it into two installments of "Z-100 Survival Kit". This issue, we'll talk about the character transmit escape sequences, and all that goes along with that. In "Survival Kit #7," we'll find out about the cursor position, and identifying the terminal type. And I'll also go into some detail about using re-directed input with a program, since these escape sequences can cause problems along those lines.

Getting Information from the Console

Usually, when you think about a computer screen, you think of an output device. You write characters or pixels to the screen, but you don't expect it to talk back. Since the display is memory-mapped, if you want to know something about what is on the screen, you simply read the contents of the video memory.

But before memory mapped displays became the rage, most computers had to make do with something called a 'terminal'. This was a separate piece of equipment which contained the video screen, and a keyboard. Input from the keyboard was received via an RS-232 line, and text output was transmitted to the terminal over the same line. Most popular terminals also had the capability to move the cursor, and perform other screen control functions, in addition to displaying text. These commands were generally prefaced by an ASCII escape character, to differentiate them from normal text.

But there was no way for the computer to tell what was displayed on the screen at any given time. Since many programs (like screen dumps, or interactive programs) needed to know what characters were in certain positions, some terminals (like the Heath H-19) had a method for overcoming this problem. An escape command was available, which when received by the terminal, caused the terminal to transmit a character (or characters) back to the computer. To the host computer, it was just like the characters were being typed on the keyboard of the terminal.

Upward Compatibility

When the Z-100 was designed, one of the major considerations was that it should be able to run 8 bit CPM software. (That was before MS-DOS had such a firm foothold). And most of that 8 bit software owned by Heath computer users was designed to be run on older Heath computers, like the H-8 with an H-19 terminal, or the H-89. The only way to get around making major changes to the existing software base was by allowing the Z-100 to emulate an H-19 terminal. And that's where we get all of these escape sequences from. They were all developed for the H-19 terminal (and therefore, for the H-89 computer).

I'm diverging from the main subject a little, but I just wanted all you new kids to know why the Z-100 has some of these weird escape commands. They're just holdovers from the previous Heath computers.

How Does This Work?

All of these terminal emulation features are implemented in the MTR-100 monitor ROM program. It is responsible for translating any escape sequences sent to the console, and performing the required function.

Most of the escape functions are performed the instant they are received — in fact, most just require that one or another flag byte be changed to indicate a new condition. But the transmit character functions work a little differently.

Whenever you tell the console to transmit a character, line, or page, it simply makes a note of the request, and then returns control back to your program. The note that it makes is kept in the MTR-100 data segment, in an area called the 'Transmit Structure'. Many of you may have seen this Transmit Structure in the monitor ROM listing (included with the Z-100 Technical Manual set) and wondered what in the heck it was for. Well . . . prepare to be enlightened.

The MTR-100 Transmit Structure is located at offset 2E7H of the ROM's data segment. (This offset will be different in older versions of the ROM.) Here's what it looks like . . .

```
XMT_STRUC
    BURST         DB    ?       ; Characters to transmit per VSYNC
    BCOUNT        DW    ?       ; Remaining characters in current burst
    COUNT         DW    ?       ; Characters left to transmit
    COL           DB    ?       ; Horizontal column to transmit
    ROW           DB    ?       ; Vertical row to transmit
    XMT_COLOR     DB    ?       ; Current color state transmitted
    XMT_GRAPHIC   DB    ?       ; Current graphic state transmitted
    XMT_REVERSE   DB    ?       ; Current reverse video state transmitted
XMT_STRUC ENDS
```

Let's assume for a moment, that you send the escape sequence ESC – (transmit current line) to the console. Regardless of whether the command was sent using BASIC, assembly language, the DOS command line, ór whatever, control is eventially routed to the monitor ROM program. The ROM program makes a note of which line the cursor is on, and saves this information in the ROW variable of the transmit structure. It puts a zero in the COL variable to indicate that the transmission should start with the first character in the line. And finally, it sets the COUNT variable equal to 80, indicating that the entire 80 column line will be transmitted. Then, control returns to your program.

If you're trying to figure out what's happening here (by disassembling the ROM code) as I have done, you're now left with a puzzle — because no characters got transmitted. A few variables were changed, but how does that accomplish anything? The answer lies in the fact that the characters will be transmitted later. All the escape sequence did was cause the ROM program to set up the Transmit Structure with instructions about what should be transmitted.

Next question. When are the characters transmitted, and by what mechanism? This takes a little more digging in the ROM code, but the solution is both clever and elegant.

The Vertical Retrace Interrupt

Some operations that affect the screen need to be done at a time when the screen is not being refreshed. Moving the cursor is one of these operations, but the most important one is scrolling the screen. If the screen was scrolled during the time that the pixels were being written to the screen, you would get annoying screen glitches when the CRT-C start address registers are updated.

One way to avoid trashing the video is to perform the scrolling and cursor updating during the vertical retrace interval. This is the period when the electron beam is returning to the top of the screen to begin a new scan. Now to most of us, the time it takes for the electron beam to zip from the bottom to the top of the screen is just a split second. But in computer CPU time (measured in millionths of a second), there's time here to do all kinds of stuff.

In order to take advantage of this 'nap' the video takes 60 times a second, the Z-100 generates a hardware interrupt whenever a vertical retrace begins. And the MTR-100 monitor ROM has an interrupt service routine that jumps right in and starts updating the screen whenever it gets control. The first thing it does is scroll the screen, if necessary. Then it checks to see if the cursor needs to be moved. And finally . . . yes, you guessed it

Continued from Page 42

There are different ways to send escape characters to the screen. These ways are explained in other bulletin board items. One way is through the prompt command (please see related bulletin board item on the prompt command). The proper syntax for a color escape sequence is:

```
ESC[#;...;#m          for MS-DOS 2.1 and 3.1
ESC[Ps;...;Psm        for MS-DOS 3.2
```

The "#" or "Ps" represents a number given in the table in the MS-DOS manual under the ANSI driver. The ";" acts as a separator for the numbers. The "m" ends the escape sequence.

An example for using an ANSI escape sequence with the prompt command is:

```
A>prompt $e[34;47m
A>prompt
A>cls
```

This would put a blue foreground onto a white background. The cls clears the entire screen so that it will be blue. The above commands could be put into a batch file. For instance, if they were put into your AUTOEXEC.BAT file, then at boot-up time you would see those colors on your screen.

*Note: Once you go into an applications program, your screen may not return to the colors you set at the operating system level. Some applications do not return the screen the way they found it. ✱

. . . it transmits any characters that need to be transmitted. But since the vertical retrace period is somewhat limited, only a few of the characters are transmitted at a time — typically 16 characters at a shot. This transmission of a small group of characters is called a 'burst'. The exact number to be transmitted during each retrace interval is held in the variable BURST in the Transmit Structure. The variable BCOUNT keeps track of how many are left in the current burst.

Accessing Video Memory

All of this might seem fairly logical, if not just a little complicated, so far. The monitor program records the request to transmit characters, initializes the Transmit Structure, then transmits the characters during vertical retrace time in order to prevent any screen interference. Right? Not quite.

That's what I thought at first too, but there's a catch. All that's required in order to transmit a character, is to read the video memory and put the character in the keyboard buffer. Well, putting the character in the keyboard buffer obviously doesn't interfere with the screen, and neither does reading the video memory.

True, in some PC compatible computers (like the original true blue IBM-PC), indiscriminate reading or writing to the video memory would cause video interference. But us Z-100 owners are lucky, because we don't have that problem. The reason we don't is because the CRT-C controller has priority over the CPU when it comes to video RAM access. Basically, this means that if your program tries to access video RAM at the same time as the CRT-C controller (for screen refresh), the CRT-C controller tells your program to take a hike until the video refresh is done. In technical terms, the video circuitry issues wait states to the CPU.

So if reading characters from the screen doesn't cause any video glitch problems, why does the ROM wait until vertical retrace time? Why doesn't it simply transmit them all at once when the escape command is received? Good question.

Searching for a Motive

As long as we know how the transmit character routines work, I don't guess it matters a whole lot why they do it the way they do. But whenever I get involved in figuring out how something works, it bothers me if I can't see the underlying logic. The situation I have been describing here is puzzling, but the answer, once known, is ridiculously obvious.

Whenever your program asks the console to transmit some characters, it must read those characters as they are transmitted. You do this just as if you are reading from the keyboard. Here's an example of how you might read a line from the console with a BASIC program . . .

```
100 INPUT"Enter line number to read"; L                ' get line number
110 LOCATE L, 1                                        ' make this the current line
120 PRINT CHR$(27);"^";                                'transmit current line
130 WHILE I$<>CHR$(13):I$=INPUT$(1):L$=L$+I$:WEND' read the transmitted chars
```

But if the monitor ROM program simply transmits all the characters at once when the escape command is received, how is your program going to gain control to receive them? In other words, by the time line 130 receives control, all the characters would have been transmitted. And the LINE INPUT statement would be waiting for nothing.

In order to make this character transmission scheme work, the monitor program needs to send a character, and then turn control back to your program, so it can read the character. Then another character can be sent, and another, until the transmission is complete. Actually, more than one character can be sent at a time, since the BIOS maintains a type-ahead-buffer which will hold quite a few characters.

Is it beginning to be obvious why the ROM program uses the vertical retrace period to send just a few characters at a time? It doesn't have anything to do with video timing. The retrace interrupt just happens to be a convenient way of transmitting the characters, while allowing your program to keep control so they can be read.

Receiving the Transmission

Okay, I can see most of you have had enough of this theoretical stuff. Time to put this information to use. The BASIC program example above gives you an idea of how simple it is to read a line from the screen, using the 'transmit current line' escape sequence.

Here is a BASIC subroutine that could be called from your program to read a line from the screen. It is similar to the program above, but variable L should be set to the proper line number before calling the subroutine. The characters from the line are returned in the string L$.

```
1000 LOCATE L, 1
1010 PRINT CHR$(27);"^";
1020 WHILE I$<>CHR$(13):I$=INPUT$(1)
                               :L$=L$+I$:WEND
1030 RETURN
```

Here is a similar routine in the 'C' language . . .

```
char linebuff[255];

getline(line)
int line;
    {
    char *buff;
    printf("\33Y%c \33^", line+32);
```

```
    buff = linebuff;
    while(*buff++ = getch()) != 13);
    }
```

And here it is again, in assembly language . . .

```
DATA     segment
HEADER       db    255,0                                  ; needed for DOS function 10
LINEBUFF     db    255 dup(?)
ESCTEXT      db    27,'Y ',27,'^','$'
DATA     ends

GETLINE proc
        mov      di, offset ESCTEXT
        add      al, 32                                  ; line number is passed in
        mov      [di+2], al                              ; register AL
        mov      dx, di
        mov      ah, 9
        int      21H                                     ; send commands to console
        mov      dx, offset HEADER
        mov      ah, 10
        int      21H
        ret
GETLINE endp
```

There's More Here Than Just Characters

If you write a program that uses one of these routines, and then run the program, you may be surprised at the contents of the transmitted line. There could be more in the line buffer than just ASCII text. So it's time we discussed the format used for the character transmission.

The first thing to realize is that characters on the screen are more than just ASCII codes. They can also be displayed in different colors. And they can be normal text or graphics characters, as well as normal or reverse video. When the ROM program transmits characters from the screen, it wants to make sure you know about the attributes of the characters, as well as the ASCII codes.

It does this by transmitting escape sequences along with the characters, in exactly the same way you use escape sequences to display characters with special attributes. The character transmission routine keeps track of the text color, graphics mode, and reverse video mode. If any changes occur in the color or modes, the appropriate escape sequence is transmitted to indicate the change. At the start of the transmission, the color is assumed to be white text with a black background, graphics mode is assumed to be off, and reverse video is assumed to be off.

Let me give you an example. The screen line we want to transmit consists of the word "ONE" in green text, followed by the word "TWO" in white text, followed by the word "THREE" in white reverse video. All three words are separated by a single space. Here are the codes that will be transmitted . . .

```
<ESC> m 4 0 O N E   <ESC> m 7 0 T W O
                  <ESC> p;T;H;R;E;E;<ESC>;q
```

. . . followed by 67 spaces, and a carriage return.

The idea here is that if you transmitted this exact string of characters back to the console, the characters would appear just as they did originally. There are only five escape sequences that will ever be transmitted from the console;

```
ESC m <fore> <back> change the foreground
                   and background colors
ESC p      enter reverse video mode
ESC q      exit reverse video mode
ESC F      enter H-19 (block) graphics mode
ESC G      exit graphics mode
```

Every transmission (even transmit character) is terminated with a carriage return. This is necessary since there is no way to tell in advance how many characters may be transmitted. Consider for instance, that the 'transmit character at cursor' command could result in a ten byte string of characters, if the character is a non-white, reverse video graphics character.

Special Problems Using BASIC

If you are using the BASIC programming language, be forewarned that BASIC plays all kinds of games with the escape sequences which are transmitted. The exact rules to the games will depend on which version of BASIC you are using.

For instance, ZBASIC version 1.x filters out the <ESC> character from the transmission, but leaves the scape sequence operands. In other words, if there is a color change, the <ESC> character won't make it, but the 'm' and color numbers do.

If you are using GWBASIC v2.x, the <ESC> characters come through, but instead of being CHR$(27), they show up as CHR$(1). Strange!

One way to avoid this situation, if all you want is the straight text (without attributes) is to use a LINE INPUT statement, instead of reading each character individually. Like this . . .

```
1000 LOCATE L, 1
1010 PRINT CHR$(27);"^";
1020 LINE INPUT""; L$
1030 RETURN
```

This is actually a much simpler way of doing it . . . and the LINE INPUT statement will strip out everything except the text characters themselves.

Buffer Considerations

One of the problems you will have to consider when writing a program that uses the transmit character escape sequences (especially transmit page) is how large to make the buffer which will hold the characters. There are 1920 text positions on

```
#define YES      (-1)
#define NO       0
#define ESC      27

struct {                                /* global character info structure      */
    char value;                         /* ascii value                          */
    char fore, back;                    /* foreground, background colors        */
    char gmode, rmode;                  /* graphic flag, reverse video flag     */
    } chr;

scrnchr(row, column)
int row, column;
    {
    char ch, buff[11], *bp;
    chr.fore = chr.back = '7';                   /* assume default values      */
    chr.gmode = chr.rmode = NO;
    printf("\33Y%c%c", row+32, column+32);       /* position cursor            */
    printf("\33_");                              /* transmit char command      */
    bp = buff;
    while((*bp++ = getch()) != 13);
    bp = buff;
    while(*bp++ == ESC) {                        /* translate escape seq s     */
        switch(*bp++) {
            case 'm': chr.fore = *bp++; chr.back = *bp++; break;
            case 'p': chr.rmode = YES; break;
            case 'F': chr.gmode = YES; break;
            }
        }
    chr.value = *(--bp);                         /* and get ascii code         */
}
```

Listing 1
'C' Function to Read Character at Cursor

```
CODE     segment
         assume cs:CODE

XMT_CMD db       27,'#$'          ; escape sequence to transmit page
BUFFPTR dw       offset BUFFER    ; pointer to current character position
BUFFER  db       2000 dup(?)      ; buffer for transmitted characters
DONE    db       0                ; flag indicating transmission is complete

GETPAGE proc near
        mov      ax, 0                   ;
        mov      ds, ax                  ;
        mov      bx, 3FEH                ;
        mov      ds, [bx]                ; get MTR-100 data segment
        push     ds                      ; save for later
        mov      bx, 83H                 ;
        push     [bx]                    ; save BIOS routine address
        push     [bx+2]                  ;
        mov      [bx], offset STUFF      ; patch in our routine
        mov      [bx+2], cs              ;
        push     cs                      ; set DS = CS
        pop      ds                      ;
        mov      dx, offset XMT_CMD      ; output transmit page command
        mov      ah, 9                   ;
        int      21H                     ;
PS2:    cmp      DONE, 0                 ; wait until characters received
        je       PS2                     ;
        pop      ds                      ;
        pop      [bx+2]                  ; restore system S_XMTC address
        pop      [bx]                    ;
        ret                              ;
GETPAGE endp

; THIS IS OUR ROUTINE TO HANDLE THE TRANSMITTED CHARACTERS

STUFF   proc far
        push     di                     ;
        mov      di, cs:BUFFPTR         ; get current buffer position
        cmp      di, offset DONE        ; out of buffer space?
        je       STUFF1                 ; yep, bail out
        mov      cs:[di], al            ;
        inc      di                     ; ready for next character
        mov      cs:BUFFPTR, di         ;
        cmp      al, 13                 ; end of transmission?
        jne      STUFF2                 ;
STUFF1: mov      cs:DONE, 0FFH          ;
STUFF2: pop      di                     ;
        ret                             ;
STUFF   endp

CODE    ends
```

Listing 2
Assembly Language Routine to Transmit Page

the screen, but in a worst case example, the transmit page routine could send back as many as 19,200 characters (ten for each text character). Of course, that's not very likely, but where do you draw the line? Is 2000 bytes enough? 3000 bytes?

My suggestion would be to do one of two things. If the character attributes are important to your application, then make your best guess about how many bytes will be transmitted for a typical screen. Be liberal, unless you're really running tight on memory. Then, design your character input routine to check for buffer overflow. It can then either discard the extra characters, or generate an error message.

The other way to handle the problem is to have your character input routine filter out the escape sequences (like the BASIC LINE INPUT statement does). This way you will know exactly how many characters to expect. Transmit character will be one byte, transmit line will be 80 bytes, and transmit page will be 1920 bytes. And, of course, each of these will have the obligatory carriage return on the end, which can also be filtered off.

Let's Take it a Character at a Time

Probably the most useful of the character transmit escape sequences is the one that transmits the character at the current cursor position. It's not often that a program would need to know every character on the screen, but it's easy to think of reasons you would want to read a single character at a specified location.

BASIC already has the SCREEN function (not to be confused with the SCREEN command) which reads the character at the cursor position. So there's no sense in showing you how to use the 'transmit character at cursor' escape sequence in BASIC. But the 'C' language has no such function (none that will work on the Z-100, that is). Listing 1 is a 'C' function that will read the character at the cursor position, along with its attributes. The character value, colors, and modes are returned in an external data structure.

A Page Full of BEEPs

If I asked for a show of hands from the people who have (on their own) been able to successfully use the 'transmit character at cursor' and 'transmit line' escape sequences, I expect there would be quite a few. But I'm willing to wager that almost no one has been able to get 'transmit page' to work correctly. Am I right? Be honest now.

When I first started playing around with it (having already mastered the character and line routines), I got some unexpected results. It seems like all you would have to do is expand the 'transmit line' routine to read about 2000 characters, and you'd have it. But such is not the case. Everytime I tried to transmit the page, I would just get a long BEEP, and the characters read from the console were all goofed up. The first couple of lines were okay, but then all kinds of characters would come up missing. The rest of the transmission was useless.

To make a long story short, I hacked the code for quite a while, trying to find out what was happening . . . in particular, what was causing that annoying BEEP. When I finally found the answer, it was something I should have suspected all along. It's amazing how clear things become after you figure them out!

The long BEEP you hear whenever you send the ESC # (transmit page) command to the console, is just the BIOS telling you that the type-ahead-buffer can't hold any more characters. The bad news is that the ROM program is transmitting the characters faster than they can be read from the type-ahead-buffer. There doesn't seem to be any way to read the characters fast enough (not even with assembly language), so the slowpoke that's causing the problem must be the DOS keyboard input routine.

Ah . . . someone is asking "How come this problem doesn't occur when a line is transmitted?". Well, you can thank the type-ahead-buffer for that. The buffer (in a Z-100) is longer than a line. Once all the characters are in the buffer, then they can be read at any speed that is convenient. But, when the ROM program tries to transmit a full page, the buffer quickly fills up, because the characters aren't being read fast enough.

The good news is that there is a solution to this problem. Actually, I can think of two ways of handling the situation.

```
CODE     segment
         assume cs:CODE
         org 100H

START:   mov      ax, 0                   ;
         mov      ds, ax                  ;
         mov      bx, 3FEH                ;
         mov      ds, [bx]                ; get MTR-100 data segment
         mov      bx, 2E7H                ; offset of BURST variable
         mov      byte ptr[bx], 8         ; (must be MTR-100 v2.x)
         push     cs                      ; set DS = CS
         pop      ds                      ;
         mov      dx, offset XMT_CMD      ; output transmit page command
         mov      ah, 9                   ;
         int      21H                     ;

         mov      di, offset BUFFER       ; READY TO READ TRANSMITTED CHARACTERS
         mov      cl, 0                   ;
PS2:     mov      ah, 7                   ;
         int      21H                     ; get a transmitted character
         cmp      al, 13                  ; end of transmission?
         je       PS6                     ; yes, skip ahead
         cmp      al, 1BH                 ; escape character?
         jne      PS3                     ;
         mov      cl, -1                  ; yes, flag to look for operand
         jmp      PS2                     ;
PS3:     cmp        cl, -1                  ; look for escape operand?
         jne      PS4                     ;
         mov      cl, 0                   ;
         cmp      al, 'm'                 ; yes, if an  m , skip next two  chars
         jne      PS2                     ;
         mov      cl, 2                   ;
         jmp      PS2                     ;
PS4:     cmp      cl, 0                   ; skip this character?
         je       PS5                     ;
         dec      cl                      ; yes, decrement skip count
         jmp      PS2                     ;
PS5:     mov      [di], al                ; save character
         inc      di                      ; bump pointer
         jmp      PS2                     ;

PS6:     mov      si, offset BUFFER       ; READY TO PRINT
         mov      cx, 24                  ; will print 24 lines
PS7:     push     cx                      ;
         mov      cx, 80                  ; 80 columns per line
PS8:     lodsb                            ; get character
         mov      dl, al                  ;
         mov      ah, 5                   ;
         int      21H                     ; print it
         loop     PS8                     ; next character
         mov      dl, 13                  ;
         int      21H                     ; print CRLF at end of line
         mov      dl, 10                  ;
         int      21H                     ;
         pop      cx                      ;
         loop     PS7                     ; next line

         int      20H                     ; return to MS-DOS

XMT_CMD  db       27,'#$'                 ; escape sequence to transmit page
BUFFER:                                   ; buffer area is in memory above program
CODE     ends
         end      START
```

Listing 3
PS.COM — A Print Screen Program which Uses Transmit Page

The first way is to simply bypass the BIOS routine, and have the ROM program transmit the characters right to your program's buffer. Each character that is transmitted from the screen is passed in register AL to a BIOS routine whose address is stored at offset 83H in the MTR-100 data segment. This address is stored in four bytes (two words); the first word is the offset, and the second word is the segment. Listing 2 shows how this might be done in assembly language . . .

Of course, this routine is designed to be included in a program. It is simply a subroutine that allows you to read the screen. One important thing to note is that the STUFF routine (our substitute for the BIOS routine) must be declared as a FAR procedure. This enables the assembler to generate the correct far return instruction needed to get back to the calling routine in the ROM program.

Now that I've made you suffer through that, I'll tell you about the other method of making 'transmit page' work correctly. This way is actually much simpler, and is probably the way the designers expected the situation to be handled. We'll just slow down the transmission rate of the characters so our normal keyboard input routine can keep up. That sounds pretty logical, doesn't it?

Remember way back toward the beginning of the column, when I was boring you to death with all the details of how the character transmission scheme worked? All that stuff about Transmit Structures, and Bursts, and such? Well here's a case where it helps to know the method behind the madness. How about if we told the ROM program to transmit less than 16 characters per burst? (Remember, a burst of 16 characters are transmitted during each vertical retrace period.) If we set the BURST variable for, say a 2 character burst, then only two characters would be transmitted during each vertical retrace. This works out to 120 characters transmitted per second, instead of the normal rate of 960 characters per second. And guess what? It works beautifully!

Listing 3 is a complete assembly language program which will dump the entire screen to your printer (PRN device). It uses the slow-down technique discussed above. You'll note that we are using a BURST value of eight. You may have to experiment with different values in your own programs to see how fast you can allow the characters to be transmitted. Using interpreted BASIC, for instance, you'll be lucky to use a BURST of 3, and get away with it. Notice also, in Listing 3, that the program filters out all the escape sequences from the transmission, since most printers would choke on them.

This program may not be much good for anything, since we normally think of a print screen program as being memory resident. (Although at 110 bytes, I believe it's probably the shortest print screen program around.) But the listing shows how you can use the transmit page feature in your own program.

Until Next Time

Well I've done used up all my space again! Remember, next installment we'll be discussing some more of those mystifying escape sequences. And I'll also be touching on the subject of using redirected input to automatically execute a program from a script file. And hopefully, there will still be a little room for a small Q & A section. 'Till then . . . keep in touch!

Drive Substitution for WordStar 3.30 on the Z-100

David S. Price
PO Box 102
Clarendon Hills, IL 60514

Mr. Dave McCrady's "Buggin' HUG" letter printed in September's REMark, describing a method for running WordStar 3.30 from any subdirectory, works great on the IBM-PC. I started using it immediately on my PC at work for editing batch and other ASCII files; I haven't used EDLIN since!

However, when I tried it on my Z-100 at home, WordStar 3.3 (referred to henceforth as WS) still had problems finding its message and overlay files. I found that the Z-100 WS version was slightly different, but was able to patch it to allow drive substitution. What started out as a letter to "Buggin' HUG" quickly grew into this text - more appropriate as an article.

WordStar Drive Specification

There is an obvious difference in the PC and Z-100 installation programs, WINSTALL and INSTALL, presenting only a small hitch. The PC WINSTALL program allows you to specify the drive where WS should look for its system files, but the Z-100 INSTALL program does not. I quickly referred to my ever-handy WS customization reference, "The WordStar Customizing Guide", by Stuart Bonney (Wordware Publishing, Inc., Dallas), which lists the DEBUG patching addresses for every internal WS variable and table. Since the Z-100 version of WS was "boiler-plated" off the IBM version (which was boiler-plated off the CPM version), virtually all of the DEBUG addresses are identical.

When the PC WINSTALL program modifies the default drive for the WS program files, all it changes in WS.COM is one byte value located at DEBUG address 02DC. If you examine the example dump of this area (provided below), you find the byte value at 02DC is 01, signifying drive A: (02=B:,03=C:,etc.) On either the PC or the Z-100, editing the byte at this address to 04 causes drive D: to be used. The DEBUG commands for dumping and editing are shown:

```
-D29Ø 2DF
xxxx:029Ø  ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ-ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ    ................
xxxx:02AØ  ØØ ØØ ØØ ØØ E9 D9 51 9Ø-9Ø C3 ØØ ØØ ØØ ØØ ØØ ØØ    ......Q.........
xxxx:02BØ  ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ-ØØ ØØ ØØ 9Ø C3 ØØ 9Ø C3    ................
xxxx:02CØ  ØØ 9Ø C3 9Ø 9Ø C3 ØØ ØØ-ØØ 9Ø 9Ø C3 ØØ ØØ ØØ Ø1    ................
xxxx:02DØ  Ø1 Ø8 ØØ Ø1 ØØ Ø7 FF ØØ-Ø1 Ø1 ØØ FF Ø1 14 ØØ ØØ    ................
-EØ2DC
xxxx:Ø2DC  Ø1.Ø4
-W
Writing 7341 bytes
-Q
```

After making and saving this change (with the DEBUG 'W' command), I then substituted a drive name for the subdirectory name as Mr. McCrady suggests, by using the DOS command:

```
SUBST D: E:\WORDSTAR
```

I expected WS to now work on the Z-100 as it did on my PC at work - NO SUCH LUCK !

Additional code in Z-100 WordStar 3.3

Zenith made a change somewhere in WS.COM that disables the use of the value stored at DEBUG address 02DC. This time luck was on my side, as I was able to find the problem fairly quickly. My WS customizing reference came in handy again, by listing the WS.COM pointer address for the "jump-to-terminal-initialization" subroutine. This is the area of WS.COM provided by MicroPro for computer manufacturers to put their machine-specific set-up routines. (For example, Zenith used such an area to set up the function key and keypad commands.) By 'U'nassembling the code in this area, I found the following section:

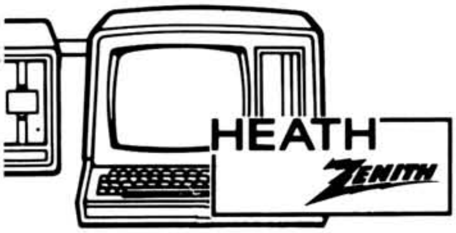
```
xxxx:55BA B419          MOV    AH,19
xxxx:55BC CD21          INT    21
xxxx:55BE FECØ          INC    AL
xxxx:55CØ A2DCØ2        MOV    [Ø2DC],AL
```

DOS function 19h (MOV AH,19) supplies the current drive number in register AL when the DOS function request, interrupt 21 (INT 21) is called. Since DOS uses '00' to specify drive A: (01=B:, 02=C:, etc.), the number must be incremented to match the drive numbering scheme WS.COM uses; which is done by the immediately following 'INC AL' command. THEN we get to the command that causes the problem, 'MOV [02DC],AL'. Do you recognize the number in brackets? No matter what change you make to the byte value at address 02DC (using DEBUG), when you run WS.COM, the Zenith code initializes it back to the current drive! In the case of a hard drive on a Z-100, this will be byte value '05', specifying drive E:.

The patch to prevent this change is simple; just remove the 'MOV [02DC],AL' command, using the DEBUG assemble command, 'A', to change the three bytes representing the problem command to 'NOP' commands (No OPeration). Note in the listing that the 'MOV [02DC],AL' command is 'U'nassembled from three bytes, A2, DC and 02. The 'NOP' command requires only one byte, '90', so it has to be issued three times to "overwrite" the problem command. The listing demonstrates the replacement procedure:

```
-A55CØ
xxxx:55CØ NOP
xxxx:55C1 NOP
xxxx:55C2 NOP
xxxx:55C3
-
```

When you issue the 'A'ssemble command (including the starting address), DEBUG will present the address to accept an 8088 assembly language command, along with a cursor. Type the new command, 'NOP', and press <return>. The next address will then be presented; in this case the address will be incremented by one byte. Enter two more 'NOP' commands in the same manner. When you have entered the third 'NOP', and pressed <return>, a fourth address will be presented. No command change is desired here, so just press <return> again; you will be returned to the DEBUG prompt. I suggest that you 'U'nassemble the section again to make sure the changes have been



Related Products

Portable Hard Disk (PHD) for Laptop Computers

Chicago, Illinois — WELTEC digital, inc. of Irvine, California announced today the development of the first Portable Hard Disk (PHD) compatible with all laptop computers.

Dave Riley, WELTEC's Director of Marketing said, "It is a major technological breakthrough. The PHD has established a new interface standard for laptop computers using the RS-232 serial port. Now there is a convenient solution for users of every laptop computer without a hard disk drive."

The WELTEC PHD provides laptop computer users with the increased storage capacity of a battery powered 20 megabyte hard disk drive. It offers simple serial port interface. And, startup could not be easier, the PHD instantly boots off the supplied floppy disk. That is all it takes, no proprietary hardware is necessary.

Riley said "All peripherals that are currently being sold for use with laptop computers have a common deficiency, they all use the only expansion slot available in the laptop." The PHD leaves the slot free for more effective use with a modem or a memory expansion card.

Although the PHD is primarily targeted towards laptop computer users, it also is compatible with any desktop computer with a serial port.

With the PHD, you have total portability of your data base, a powerful advantage for any business executive who travels. The PHD will connect to the serial port of any PC, either desktop or laptop, and you are instantly on stream with your entire data base. In addition, if security is important, the highly-portable PHD allows you to take your data base with you when you leave the office.

Currently, the PHD is available with 20 megabytes of storage capacity, a 40 megabyte option will be available later this year.

For more information, contact:
WELTEC digital, inc.
17981 Sky Park Circle, Blg. M
Irvine, California 92714

made in the correct place, effectively deleting the 'MOV...' command.

If you have an older copy of DEBUG, you will not have the DEBUG 'A'ssemble command available to you. You can still make the patch using the 'E'dit command to change the three byte values to '90', like so:

```
-E55C0
xxxx:55C0  A2.90  DC.90  02.90
```

The 'E'dit command presents you with the value at the address specified, followed by a period. Type in '90', then press the <space bar>. The next byte will then be presented for editing. When you have patched the last of the three bytes, press <return> instead of the <space bar>. If you make a mistake at the last byte by pressing the <space bar> instead of <return>, don't worry; just don't type a number. Pressing <return> will kick you back to the DEBUG prompt without a change to that last byte.

Finishing up

Patching address 02DC to your desired drive, and patching the command at address 55C0 to three 'NOP' commands are the only patches required. Save your new modified WS.COM with 'W', the DEBUG write command, and exit with 'Q'. Remember, make the changes to a copy of WS.COM, never to your distribution copy.

```
-W
Writing 7341 bytes
-Q
```

Mr. McCrady's solution then works as it should. Instead of using the AUTOEXEC batch file to set up the drive substitution as he does, I use a batch file named WS.BAT:

```
ECHO OFF
CLS
SUBST D: E:\WORDSTAR
SET OLDPATH=%PATH%
SET PATH=%PATH%;E:\WORDSTAR
KEYMAC \WORDSTAR\WS.KM > NUL
WSRUN
KEYMAC \UTILITY\NULL.KM > NUL
SET PATH=%OLDPATH%
SET OLDPATH=
SUBST D: \D
```

This batch file makes the drive substitution, saves to the DOS environment space the current path (which I call OLDPATH), adds the \WORDSTAR subdirectory to the current path so DOS can find WSRUN.COM, loads the HUG Z-100 key macro program, and runs WordStar. If you don't use a key-assigning macro program, just leave out the two commands 'KEYMAC ...' Note that you must rename WS.COM (I used WSRUN.COM) to prevent DOS from running WS.COM when you wanted to run WS.BAT. As long as WS.BAT is in a subdirectory pointed to by the current DOS path (mine is in E:\BATCH), you can run WordStar from ANY subdirectory, and WordStar will present for editing a directory listing of files IN THAT SUBDIRECTORY. After exiting WordStar, this batch file restores the original system path and removes the substitution of drive D: to prevent drive complications with applications other than WordStar (FORMAT and DISKCOPY can be particularly dangerous.) The drive and path changes are thus in effect only when you need them!

Additional Advice

The above batch file also demonstrates how the DOS environment space can be used to save and restore the path. The DOS environment is a broad topic in itself, and can be useful and interesting to explore. REMark and other computer publications (particularly PC Magazine), periodically publish articles or letters regarding the benefits of using the environment in your batch files. If you want to look at what's currently in your environment, type at the DOS prompt:

```
SET <return>
```

If you have a hard disk, and you are not using batch files to change subdirectories and run your applications, you can save yourself a lot of typing by letting a batch file do the repetitious work for you. My batch file subdirectory is full of such, and I have access to them from any other subdirectory because E:\BATCH is in the PATH established by my AUTOEXEC.BAT file.

I encourage other Z-100 users to contribute articles, tips and suggestions to REMark, no matter how simple or insignificant they may seem. You don't have to be a journalist or be proficient in prose to provide assistance to others. Don't exclude older software topics either. I love to see an older program like WordStar 3.3 remaining the topic of so much discussion and interest, and its users' preference over newer programs. I hope these patches make WS 3.3 more useful and longer-lived for other loyalist Z-100 HUGgies.

Great Inexpensive Software for the Z-100

Gregory D. Elder
Qtrs 4301-A
USAFA, CO 80840

The February 1988 issue of REMark contained an article entitled "$100 (or less) Software for the Z-100." It described various public domain, shareware, and commercial software which runs on the Z-100 with ZPC. However, many Z-100 owners may not be aware of the fact that a wealth of inexpensive software exists which runs on the Z-100 in native mode (i.e., without the need for ZPC). I'll admit, the amount of software for the Z-100 isn't as great as that for the IBM-PC. But even though Zenith has discontinued the manufacture of the Z-100, it still is one super computer. Enough Z-100 specific software does exist to perhaps make your use of the Z-100 easier and more efficient.

In this article, I will describe software which has been useful to me or fun to use (we can't forget those computer games). Most of this software I have obtained through local Heath/Zenith user groups or Z-100 specific bulletin boards. All of the software mentioned in this article is either public domain or shareware. Public domain software is free for you to use as you wish (with the exception of selling it). On the other hand, shareware (or user supported software) is not free. Normally, you can try out shareware programs for a limited time. If you like the program and want to keep it, you must send in the requested registration fee to the author. Be sure to read the documentation which comes with the software to determine if it is public domain or shareware. One final note — this article does not provide an in-depth review of each software product. It simply describes the software and lists some of the major features.

Utility Programs

BESTERM

Years ago when I first began using a Z-100, the only communications software I had to use with it was Modem7. Of course, this granddaddy of communications packages only ran under CP/M. Therefore, when downloading MS-DOS files from bulletin boards I would first have to capture the files under CP/M, and then use the RDCPM program under MS-DOS to copy the files to an MS-DOS formatted disk. This process became tedious rather fast.

Eventually, a great communications package became available for the MS-DOS side of the Z-100. This public domain program is called BESTERM. It doesn't have all the bells and whistles of many of the commercial comm packages you may have seen advertised for IBM-PCs (like CROSSTALK or HyperAccess). On the other hand, it is a simple program which does not require you to wade through a 100+ page user's manual in order to understand how to use it. BESTERM has easily met most of my communications requirements. It of course has a terminal emulation feature to allow you to communicate with other computer systems. In addition, BESTERM allows you to upload and download files. It supports both the Xmodem and KERMIT file transfer protocols. BESTERM also makes full use of the Z-100's function keys for program control. One feature which I really like is the phone book. A phone book is a free formatted file you create with your favorite editor. The phone book contains a directory of BBS phone numbers. Pressing the F1 key in BESTERM will display the phone book on your screen.

You can then press a single key to have BESTERM automatically dial the appropriate phone number. The current version of BESTERM (2.8) requires MS-DOS 2.0 or above and uses about 70K of memory. For a free communications package, you can't beat BESTERM!

CED100

CED100 is a Z-100 version of this super command editor. Features of CED100 include improved command line editing, recall of previously issued commands, and command synonyms just to name a few. CED100 requires MS-DOS 2.0 or later. Once CED100 is loaded, it remains resident in memory and becomes an extension to DOS (i.e., CED100 looks like a built-in DOS command).

I think CED100's best feature is its command line editor. Unlike the DOS editor, CED100 allows you to move the cursor around within a command for corrections and alterations. For example, suppose you type "DIR A:*.DOC". Before you press the RETURN key to process this command, you realize you really wanted to look at the DOC files on drive E. With CED100 installed, you can simply use the arrow keys to position the cursor over the "A" and then replace it with an "E". By the way, the <I Chr> key toggles between type over and insert mode for CED.

CED100 also keeps track of the MS-DOS commands you enter. The commands are stored as a stack and you can retrieve the commands by using the up and down arrow keys to redisplay them. For example, suppose you executed the following three commands:

```
COPY A:*.DOC B:
DIR B:
REN B:TEST.DOC B:TEST.TXT
```

Now if you want to execute the first command (COPY A:*.DOC B:) again, you simply press the down arrow key. CED100 will display the copy command and you can then execute it by pressing the RETURN key. Each time you pressed the down arrow key, CED100 would display the next command stored in its stack.

The last CED100 feature I wish to mention is synonyms. CED100 allows you to define synonyms for commonly used commands. For example, suppose you frequently copy DOC files from drive A to

B. You could define a synonym for that command by typing the following:

```
CED SYN cpdoc copy a:*.doc b:
```

Now every time you type "cpdoc", you will actually execute "copy a:*.doc b:".

CED100 version 1.0 is in the public domain. In the documentation, the author mentions making some enhancements to CED for the IBM-PC and selling the new version commercially. The author had no plans for a commercial version for the Z-100. The first time I used CED, I wondered how I had gotten along without it. This is one program I have added to my AUTOEXEC.BAT file to make my use of the Z-100 much easier.

KERMIT

No list of great, inexpensive software would be complete without mentioning KERMIT. KERMIT is a communications program which has been around almost as long as Modem7 (to be honest, I don't know which program was first). The beauty of KERMIT is that versions exist for just about every type of computer system (micros, as well as mainframe computers). For examples, versions of KERMIT are available for Digital Equipment Corp. (DEC) VAX computers, IBM mainframes, Apple IIs, IBM-PCs, Macintoshes, and of course, Z-100s. KERMIT has its advantages if you work in an environment with many different computer systems and have a need to transfer data between the various systems. I used to work in such an environment, and believe me, KERMIT made my job of moving files between a VAX, DEC-2060 mainframe, and Z-100 very easy.

Like most communications programs, KERMIT provides terminal emulation to allow you to access other computer systems. The Z-100 version of KERMIT emulates a DEC VT52 terminal. In addition, KERMIT has the capability to transfer files between systems using its own file transfer protocol. If you like, you may also send files using a "raw" transfer (i.e., without the error checking provided by the KERMIT protocol). One feature I really like in KERMIT is file name collision avoidance. If the file you are transferring to your Z-100 has the same name as a file already on disk, KERMIT will not overwrite the existing file. Instead, KERMIT will rename the new file.

KERMIT also allows you to create command macros and script files. A command macro is a single command word which represents several other commands. For example, in KERMIT, you could define a macro called "setup" to set your comm port to J1, the baud rate to 4800, and parity to even by giving the following command:

```
define setup set port 1, set speed 4800,
  set parity even
```

When you want to issue the commands contained in setup, you simply invoke the command "do setup." Command macros are limited to 128 characters. Therefore, if you want to define something larger, you can place the commands in a script file. This is simply an ASCII text file containing all of the commands you want to execute. When you want to process those commands, you simply invoke the command "take <filename>", where filename is the name of your script file.

LOOK

Look is a file browsing program for the Z-100. You can always use the MS-DOS TYPE command to view a file on the console, but you have to be fast with the Control-S if you want to pause a particular part of the file on your screen. Also, TYPE does not allow you to move backwards in a file. Therefore, in order to view part of a file a second time, you must retype the file from the beginning.

Look is a bi-directional file viewer. You can move forwards or backwards through a file. Look also lets you view the file one line at a time or one screen at a time. Furthermore, Look allows you to specify up to 10 files for viewing and allows you to switch between files with one or two keystrokes. Here's how you would start Look to be able to switch between 3 files.

```
LOOK FILE1.EXT FILE2.EXT FILE3.EXT
```

Look will begin by displaying FILE1. To view the next file, you type 'N' for Next File. Look will then prompt you with FILE2 and ask if you want to view that file. If you answer yes, FILE2 will be displayed. If you answer no, Look will prompt you for FILE3. Look maintains a circular list of the file names you entered on the command line when you started the program. Also, when you switch between files, Look will remember your location in the first file. Then if you switch back to the first file, you will be viewing it from where you left off.

Here are a few other features of Look. Look can scroll horizontally to view files wider than 80 characters. In addition, Look has a continuous scroll mode with variable speed to let you continuously view a file line-by-line. Look version 1.2 is in the public domain. (By the way, I wrote Look myself as a way to get practice programming in Modula-2. I also wanted to contribute something to the public domain in return for all the great software others have written). In case you have an IBM-PC computer, an IBM version of Look is also available.

MicroEMACS

MicroEMACS is an excellent text editor written for a variety of microcomputers. It is based on the original EMACS text editor written by Richard Stahlman at MIT in the early 1970s for Digital Equipment computers. If you ever worked with a PDP-11 computer or UNIX, you have probably come across EMACS.

The Z-100 version of MicroEMACS is a very full featured text editor. It allows you to create and modify documents, programs, and other text files. It is powerful and yet easy to use. MicroEMACS provides all the editing features you would expect, such as deleting characters, words, blocks of text; string searching and replacing; paragraph reforming; and case controlling (switching current text to upper- or lowercase letters).

MicroEMACS also has some special features not found in some commercial editors. For example, MicroEMACS allows the screen to be split into different windows to allow you to edit more than one file at a time. It comes in handy when you want to copy or move text from one file to another. Another powerful feature of MicroEMACS is macros. Macros are programs used to customize the editor or to perform complicated editing tasks. If there is a particular sequence of commands you use frequently when editing, you can store those commands in a macro file or buffer. The macro can then be bound to whatever keystroke you prefer. Whenever you press the particular key you have chosen, all the commands in the macro file, or buffer, will be executed.

The majority of MicroEMACS commands are executed via control keys (similar to WordStar) or the meta key. On the Z-100, the meta key is the escape key. Let me give some examples of commands in MicroEMACS. To move the cursor to the beginning of a line, you type Control-A. To move to the beginning of a file, you type Meta-< (this means press the escape key and then the "<" key). The command to move the cursor up one line is Control-P, while the command to move up one paragraph is Meta-P. Using the control and meta keys for commands may seem a little awkward at first (and will take time to learn as well). However, you can bind some of your most frequently used commands to function keys to reduce the number of keystrokes you must type. Also, MicroEMACS comes with an on-line tutorial to help you learn the editor. I'll admit, I haven't taken the time to fully use and learn every feature in MicroEMACS. I would think that once you have learned all the features of MicroEMACS, you could do some "powerful editing."

By the way, there are versions of MicroEMACS for a number of different personal computers. For example, MicroEMACS is also available for the IBM-PC, HP110, HP150, Atari 520/1040ST, and Amiga 1000. (One of the reasons for using the control and escape keys for commands is to make the keystrokes for the commands the same across various machines. A save-file command is the same on a Z-100, IBM-PC, Atari, or Amiga.) One other point worth noting about Micro-

EMACS — it comes with the full C source code for the program. If you ever wondered how an editor works, you now can have access to the code and see for yourself.

VSWEEP

Those of you familiar with CP/M may have come across a program called VSWEEP. VSWEEP is a similar program written for the Z-100 and MS-DOS version 2.0 or higher. Basically, VSWEEP is a file manipulation utility which gives you a window into a disk directory. Upon execution, VSWEEP will display on the computer screen the names of the files located in the designated directory. You may then use the arrow keys to select a specific file for a VSWEEP operation (the file selected will be displayed in reverse video).

VSWEEP commands may be divided into tagging operations, file operations, directory operations, and movement operations. The tagging commands allow you to mark or unmark files for mass operations. For example, you can copy and delete single files. In addition, you can perform mass copy and delete operations on tagged files (i.e., every marked file will be copied or deleted). When performing mass file operations, you have the option of performing the operation with or without user intervention. User intervention means VSWEEP will ask you to approve each individual copy or delete command before it is made. VSWEEP also provides for renaming of files, printing files, and viewing files on the console.

The VSWEEP directory commands allow you to switch directories, show only those file names which match a particular mask (including wild cards), and display space available on a disk. The movement commands simply allow you to move the cursor among the various file names shown on the screen. For example, the left arrow key moves the cursor to the file name left of the current cursor position. The "+" key moves to the next screen (if there are more file names than will fill one screen).

Z100CAT

I have to admit that when it comes to my collection of diskettes, I am very disorganized. Many times when I want to find a neat little program I remember having used once, I begin my ritual. This consists of inserting disks into the computer one at a time and executing a directory command. On about the ninth or tenth disk (if I am lucky), I locate the desired program.

Fortunately for me, there is Z100CAT. This program allows you to organize your collection of floppy disks. Z100CAT is menu-driven for easy use. Its basic feature is to read all of the files stored on your disks and to create a file containing the volume names, file names, and an optional comment about each file. You can print this file sorted in alphabetical order by file name, volume name, or grouped by extension name. It even has an option to find duplicate file names in case you have multiple copies of the same file on different disks.

According to the Z100CAT documentation, the author has placed it into the public domain under the Shareware concept (I think this is a contradiction in terms). In any case, he asks for a $10.00 donation if you find the program useful.

ZYAC

ZYAC (Yet Another Calculator program) turns your Z-100 into a scientific calculator. It may come in handy if you need to perform some basic calculations and don't want to run upstairs (or wherever) to find your pocket calculator. ZYAC performs all the basic math functions you would expect (addition, subtraction, multiplication, and division), plus the trigonometric functions (i.e., sine, cosine, tangent, their inverse functions, and the hyperbolic functions as well). Other features included with ZYAC are the use of scientific notation for very large numbers and the following additional functions: square, square root, power, logarithm (common and natural), exponential, reciprocal, factorial, and statistics (to include mean, standard deviation, and variance). Also, ZYAC provides for some basic conversions between the English measurement system and the metric system.

As you can see, ZYAC provides all the features found in a good, scientific pocket calculator. ZYAC is an Algebraic Operating System (AOS) calculator, as opposed to a reverse Polish notation system. (Like a Texas Instrument calculator and not a Hewlett-Packard calculator).

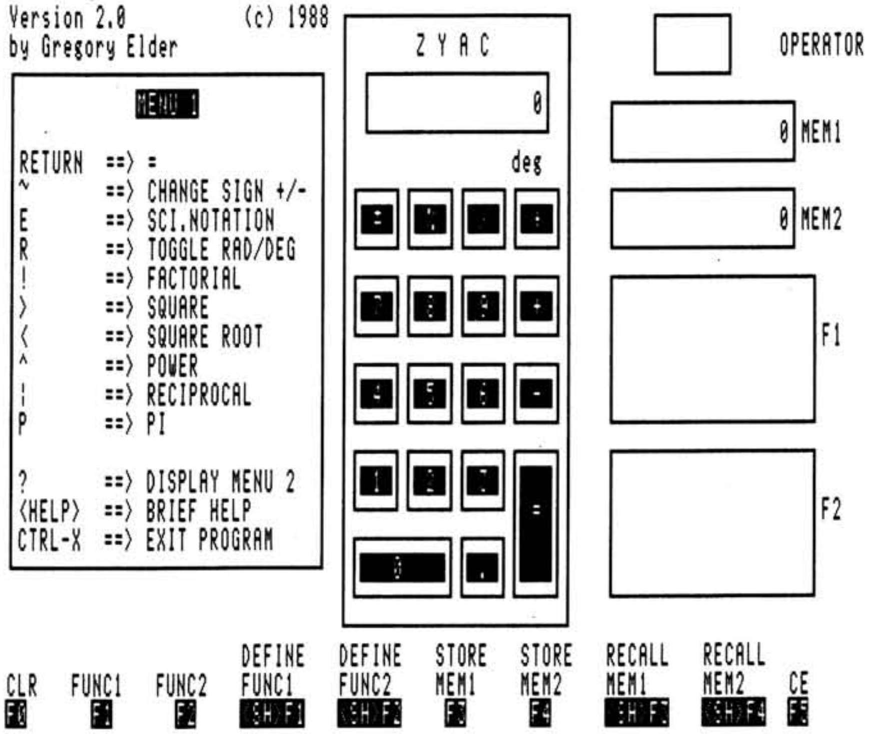
ZYAC draws the image of a calculator on the middle of the computer screen. The keypad on the Z-100 corresponds to the layout of the calculator keys shown on the screen. The left-hand side of the screen displays a menu of commands. The right-hand side of the screen displays information about values stored in memory and programmed functions. (ZYAC has two memory storage areas that allow you to save results. Also, ZYAC has two programmable function keys which allow you to program equations, formulas, etc.) Figure 1 shows what ZYAC looks like on the computer.

One nice feature of ZYAC is that stored memory values and equations for the two programmable function keys may be saved to a file. The contents of the file may then be loaded into ZYAC to initialize the program each time you run it.

ZYAC version 2.0 requires MS-DOS 2.0 or higher. It has the option to run in color if you have a color monitor. Version 2.0 is shareware and costs $15.00 to register with the author. (By the way, this is one other program I wrote myself. This is the last time I will mention software I have written.)

Games

Like most computer users, I enjoy playing computer games every now and then. (You know what they say about all



```
Version 2.0          (c) 1988
by Gregory Elder

        MENU 1

RETURN  ==> =
~       ==> CHANGE SIGN +/-
E       ==> SCI.NOTATION
R       ==> TOGGLE RAD/DEG
!       ==> FACTORIAL
>       ==> SQUARE
<       ==> SQUARE ROOT
^       ==> POWER
|       ==> RECIPROCAL
P       ==> PI

?       ==> DISPLAY MENU 2
<HELP>  ==> BRIEF HELP
CTRL-X  ==> EXIT PROGRAM
```

Figure 1

work and no play). I will briefly mention some of the games I have tried out. Most are written in BASIC and will run under ZBASIC or GW-BASIC.

Chess

Yes, there is a chess program for the Z-100. The one I have is written in BASIC. It does a nice job of using the Z-100's graphics ability to display the chess board and chess pieces. It has 24 skill levels from which you can play. I have only played a complete game using skill level one. (Surprisingly for me, I did beat the computer). With this chess program, you can save a game to disk for resumption at a later time. One draw back to the game is that it is slow (i.e., the program takes its time "thinking" its next move). The higher the skill level, the slower the game. I have only played it using the ZBASIC interpreter. I'm sure it would run faster if compiled.

Football

What to do when football season is over? Take out this BASIC program and play your own game! After picking the names for the two opposing teams, the program draws a football field on the screen. Then, a stick figure referee runs to the middle of the field, along with the team captains. A coin is then tossed to determine the receiving and kicking teams. This game even displays a score board showing current scores, period, down, and time remaining in the game.

To play the game after its initial setup, the two players simply enter their offensive and defensive plays. You have 14 offensive plays to choose from, such as end sweep, reverse, quarterback sneak, bomb, and option pass. The 6 defensive plays include 4-3 defense, prevent defense, and quarterback blitz. I suppose one problem with this game is that the offensive and defensive plays are entered at the same time. Therefore, the defensive player knows what play the offense is using before making his defensive decision.

After the plays have been entered, the computer determines the outcome. A football is then moved to the appropriate position on the field and the play continues. I suspect there may be a bug or two in this program judging from the outcome of some plays. However, you do get the source code with this game so you could make changes as you like.

Patrol/Stalker

Patrol and Stalker are two Z-100 games I downloaded from a Z-100 specific bulletin board system. Patrol is your basic space invaders clone. This program does a nice job of showing-off the graphics capability of the Z-100. You use the '4', '5', and '6' keys on the keypad to move your ship and the '0' key to fire missiles. This is a nice game.

In Stalker, you fly a helicopter out of enemy territory. You are pursued by enemy helicopters which attempt to shoot you down. The numeric keypad is used to move your helicopter, while the 'A' and 'S' keys fire missiles and drop bombs. Both Patrol and Stalker are compiled programs.

Ratrun

Ratrun is another BASIC program. In this game, you are a rat running through a maze, trying to find the cheese. At the beginning of the game, you determine the size of the maze. The program then randomly creates a maze and "places" the cheese somewhere in the maze. You are then presented with 3-D views of your current location in the maze. You use the arrow keys to move about. Of course, your objective is to locate the cheese in as few moves as possible.

Worm

Worm is another interesting BASIC game. In it, you use the numeric keypad to move a worm around the screen. Numbers appear at random locations on the screen which you direct the worm to for eating. Each time the worm eats a number, it grows longer. The worm cannot run into its tail or any of the side walls on the screen. The object of the game is to score points by having the worm eat as many numbers as it can before running into itself (or a wall).

In this article, I have described some of the public domain/shareware programs I have found useful, or fun, on my Z-100. All of these program run in native mode on the Z-100. Some of you reading this article may know of other great software. If so, why not write an article for REMark about it so other Z-100 owners can be aware of the program.

Software Discussed

BESTERM
Bruce Stock
Available on the HUG BBS as BESTERM .ARC.

CED
DOS Command Editor, Version 1.0
(Z-100 version)
Christopher J. Dunford
10057-2 Windstream Drive
Columbia, MD 21044
Available on the HUG BBS as CED100 .ARC.

KERMIT
Frank de Cruz and numerous others.
KERMIT Distribution
Columbia University Center
for Computing Activities
612 West 115th Street
New York, NY 10025

LOOK version 1.2
Gregory D. Elder
Qtrs 4301-A
USAFA, CO 80840

MicroEMACS
Daniel M. Lawrence

VSWEEP
Available on the HUG BBS as VSWEEP .ARC.

Z100CAT version 3.11
Albert Plitt
4714 Salterforth Place
Ellicott City, MD 21043

ZYAC version 2.0
Gregory D. Elder
Qtrs 4301-A
USAFA, CO 80840

CHESS

FOOTBALL, RATRUN, and WORM
Available on the HUG BBS in the file BASGAME1.ARC. This file includes other BASIC games as well.

PATROL and STALKER
Glenn Herrin

Software discussed in this article not on the HUG BBS has been provided to HUG. Perhaps the editors can publish the names of the files for those programs after they are loaded into the BBS. ✱

Desktop Publishing on a Z-248

Jim Thornberry
Public Affairs Officer
Defense Contract Administration Services Region Dallas

Desktop publishing has entered the PC market with a roar, and promises to exceed even Apple's wildest dreams.

That said, you will still find desktop publishing on your Zenith 248 a challenge. But don't despair. The following tips will help you fine tune your machine. These ideas are geared for Pagemaker 3.0 on the PC by Aldus Corp., but should be easily converted to Ventura Publisher by Xerox.

First, I assume you are using an 80286 machine with matching math coprocessor and a hard disk. If you're not, you should be. I also assume you are using a Postscript laser printer. Again, if not, you should be. Desktop publishing requires more horsepower than anything less can deliver.

The CONFIG.SYS file in your root directory (C:\) should look like this:

```
device=[path]\ansi.sys
files=20
buffers=30
shell=command.com /p/e:40
device=[path]\vdisk.sys [b] [s] [d] /e
```

The ANSI.SYS device is found in your MS-DOS files and is nothing more than a program to control screen display.

By increasing the size of the files and buffers, you allow DOS more maneuvering room, especially since Pagemaker gobbles up Random Access Memory (RAM).

The SHELL command allows you to make COMMAND.COM the only command interpreter (/p) and to expand the environment in which it operates (/e:40).

Now for the fun part, a RAM drive.

VDISK.SYS, also found among your DOS files, sets up a "disk drive" in part of your RAM. The parameters following the VDISK.SYS command control, respectively, the size of the drive (64K is the default), the sector size (128 is default) and the number of directories (64 is the default). The "/e" places the drive in that portion of your computer's memory above 640K, if you have extended memory. If you don't, leave out the "/e". If you have extended memory, designate all of it for your RAM drive. (If you have MS-DOS 3.3 or later, VDISK.SYS has been replaced with RAMDISK.SYS.)

A caveat here. Don't use your RAM drive to store or save anything, because if the power goes or you reboot your system, anything in RAM not saved to hard or floppy disk goes, too.

So what's the advantage? you ask. Well, a RAM drive offers much faster access time, since it operates at the speed of your computer's memory (which is measured in nanoseconds). This is going to be just great for a graphics program like Pagemaker, which does a lot of swapping of temporary files from disk to memory. If you tell Pagemaker (below) to use the RAM drive, the program will be doing its swapping a lot faster, and you won't be sitting there sipping coffee.

But again, remember, SAVE! SAVE! SAVE!

Now for the AUTOEXEC.BAT file, also in your root directory; you should include several items:

- A PATH statement, which should begin with d:\; (or whatever letter DOS assigns to your RAM drive); the path should also have directions to your desktop publishing directory.
- After the path statement, the following:

```
ECHO Please wait until Laser Printer is
        "Online/Idle"...then...
PAUSE
COPY [path]\psprep.txt [laser printer port]
COPY c:\command.com d:\
SET TEMP=d:\
SET COMSPEC=d:\command.com
```

Downloading the Postscript prep text (PSPREP.TXT) to your laser printer at the beginning of each work session will make Pagemaker's printing process a little faster.

Copying the COMMAND.COM file to your RAM drive (D:\, or whatever) will tell DOS to reload those portions of the file which are overwritten by some of your application programs from the RAM disk, again a little faster process.

Telling DOS to place temporary files

in the RAM drive will speed up Pagemaker (swapping from disk to memory and back, remember?) and any other program which creates temporary files as part of its backup or working environment (e.g., WordPerfect).

Telling DOS to look in the RAM drive first for the Command Specifications simply speeds up things some more.

Pagemaker runs under Windows. Changes in the WIN.INI file are in order to match the DOS operating environment.

The WIN.INI file needs the following statement in its postscript section:

```
header downloaded=yes
```

Remember, you copied PSPREP.TXT to the laser printer in the AUTOEXEC.BAT file. Now you are telling Pagemaker (through Windows) this file is already resident in the printer's memory and that it doesn't need to be reloaded. This will shave a couple of minutes off each print job.

The WIN.INI file also needs the following parameters for the swapdisk and swapsize statements in the [pif] section:

```
swapdisk=d
swapsize=256
```

A separate batch file to execute Pagemaker is useful (call it PM.BAT). It should look like this:

```
echo off
cls
cd\[directory]
win pm %1
cd\
menu
```

Install Pagemaker as a subdirectory of Windows (unless you use only the run-time version of Windows, in which case, a separate directory called PM will do just fine).

At the C:\ prompt, type pm and ENTER, and you will open Pagemaker. (If you remember the path and file name of your document, the %1 replaceable parameter in this batch file will allow you to open that document directly from DOS by typing something like pm c:\[directory]\[filename].pm3 at the prompt.

If you are using the full version of Windows, Pagemaker should be open by itself. Other applications, such as your cardfile or calendar or Windows Writer, if open, will slow Pagemaker down considerably. Remove from your WIN.INI file any comments after run: and load:. If you previously used these statements to open Windows Desktop Applications, such as your calendar or cardfile, you can open them with separate batch files and place them in your menu. The batch files could look like this:

a. For your calendar:

```
echo off
cls
cd\windows
win [name].cal
cd\
menu
```

b. For your cardfile:

```
echo off
cls
cd\windows
win [name].crd
cd\
menu
```

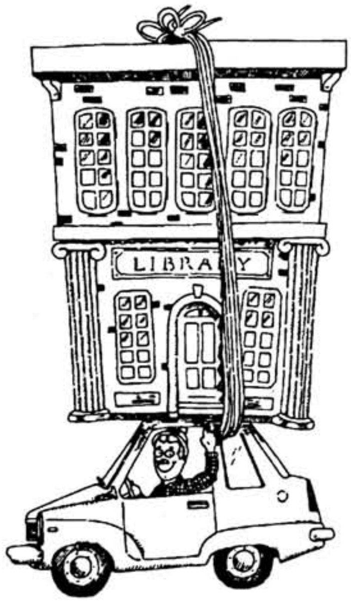
You can name the first CAL.BAT and the other CRD.BAT. Type cal or crd at your c:\ prompt and you go directly to that application.

After you have your WIN.INI file like you want it, put a read-only flag on it to prevent it from being inadvertently erased or changed.

There are other things you can do, like work with a copy of Pagemaker directly in your RAM drive. This is really super fast, but I am personally reluctant to put all my eggs in one RAM, so to sepak.

When using the above configuration on your PC, remember to save your work often. Every five minutes is not too often. The short amount of time it takes to use Pagemaker's "Save As ..." command will be many times less than the time you use to rebuild a document if your grandson crawls under your desk and pulls the plug. ✱

Library Support for C Language Applications Programming



Harold C. Ogg
357 W. Diversey Avenue
Addison, IL 60101-3508

While the C programming language can purport to attribute itself a "middle level" lexícon, often it can revert to nature and act like an "assembler in sheep's clothing." For all its capabilities, C can be gruesome for the neophyte who expects ready-made operands and macros for even the simplest of I/O or data formatting routines. Pity the unsuspecting programmer fresh from the worlds of COBOL or BASIC where the ubiquitous "prints" and "accepts" are ripe for the plucking. C, in fact, is virtually barren of all but the minimum complement of reserved words. Kernighan and Ritchie[1] lay claim to a mere 32 reserved identifiers, and the proposed ANSI standard[2] does not give evidence that an enriched vocabulary will provide the basis of expanding the language's power.

C compiler publishers have followed the route of library support to unlock the potential of the language. Indeed, the documentations abound with each company's interpretation of what constitutes the "best" method to implement a standard library function. Now passing its tenth anniversary year, the C language can boast a broad repertoire of callable routines to accomplish everything from string handling to interrupt servicing. In order to market a competitive product, the software companies take the basic library appendages and "salt" the product with some unique (and generally very clever) code handling routines of their own. A quick rundown of the listings in the libraries index of any software manual will indicate the origins of routines with UNIX, MS-DOS, iAPX86, Lattice (or other proprietary company), and XENIX, as well as the standard ANSI, to name but a few. Most of the library modules will have begun life as other C modules, using nothing more than the 32 reserved words glued together with the appropriate syntax and operators, to comprise more complex, sophisticated and powerful object code. Where C code is not sufficiently responsive or is too verbose, compatible assembly code is used to create object modules that fill the missing links.

Given the relative youth of the C language, software engineers have done a splendid job of exploiting C's possibilities. A perusal of the library source code for a given C compiler will generally prove that it is compact and addresses each of its intended functions in an efficient manner. It is the latter point that is the focus of this article. In providing a fast compiling, quick executing developmental tool rated G (general programming audiences), complexity must frequently be sacrificed. To be sure, compiler software publishers make their products attractive by including graphics libraries, windows support, coprocessor code generators, and linkable device drivers. But these accessories must remain open to a variety of interpretations, and one is still left with the prospect of reinventing the proverbial wheel each time a specific application is indicated. And it seems that specifics and custom programs are demanded about 99% of the time.

Thus enters the world of C function libraries. Often it is a task merely to establish the existence of some prewritten code that will accomplish (or nearly so) the result which you want to create. Also a function of its adolescence, the body of literature on C programming is only now coming into its own. Advertisements for library routines are often placed in obscure journals, or not even advertised at all. And there exists as well the chance event of accidentally stumbling onto an appropriate routine via a public bulletin board or through a piece of shareware. The last avenue also opens the possibility of picking up a virus or, more realistically, naively incorporating a selection of inadequately

Harold C. Ogg is Automation Librarian for Chicago State University.

debugged object code well beyond the point of no return.

When purchasing libraries, the programmer needs to act as an intelligent consumer on several matters. One should ascertain whether new code incorporating others' libraries is subject to royalties. As far as can be determined, none of the libraries in this article carry the requirement that royalties be paid, but the developmental programmer should check the license agreement before proceeding with sales. Also, a programmer should check carefully the age of the library .obj files (i.e., whether they were created with a previous, and now obsolete version of a compiler(s)). It should also be determined that .obj code for all of a compiler's memory models is supplied with the library package. It is a great advantage if the author(s) provided full source code for the libraries. But check also to see whether separate licensing is necessary if you modify or customize the original source. It is also prudent to ascertain that the library code isn't hardware specific; generally, it is better if the library developer has relied on interrupts rather than BIOS calls. And if you yourself develop machine specific code, be ethical and inform potential customers that such is the nature of your program.

A number of developmental software publishers, such as Microsoft, provide lists of third party libraries with their compiler packages. This is true of hardware manufacturers as well. Some companies are reluctant, however to give even the hint of endorsement of another firm's products, and will not recommend out-of-house software. Others (like Hewlett-Packard) attempt to work with many suppliers for mutual customer/developer benefit.

One of the most frequently asked questions regarding developmental packages is: will it support OS/2? Most quality C compilers can now break the 640K barrier with little difficulty, whether the code is concerned with EMS 4.0 support or the accessing of extended memory. It would be nice if the issue were only a matter of watching re-entrancy and thread' main-

```
/* Creates and opens a database file for names and addresses. File is indexed
   by area code. Since no math is performed on telephone numbers, type char
   will be used. Resultant file and index is usable by dBase III+. */

#include        <stdio.h>
#include        <dbc3plus.h>   /* header file for small memory model */

#define ERROR 1         /* not in header definitions--evaluates false */

char *blakbook;         /* file descriptor for BLAKBOOK.DBF */
char *areacode;         /* file descriptor for AREACODE.NDX  */
char record[80];        /* equal to total length of dBFIELD */
char codekey[3];        /* area code key buffer */

        dBFIELD flds[5] =               /* just like dBase III+ */
   {
        "NAME",     'C', 20, 0,         /* record positions [0-19]  */
        "ADDRESS",  'C', 25, 0,         /* record positions [20-44]  */
        "CITYSTZIP", 'C', 25, 0,        /* record positions [45-69]  */
        "AREACODE", 'C',  3, 0,         /* just use ascii; positions [70-72] */
        "PHONE",    'C',  7, 0          /* here, too; positions [73-79] */
   };

main()
{
if (dBcreat ("BLAKBOOK.DBF", 5, flds) != SUCCESS)
        return (ERROR);
if (dBicreat ("AREACODE.NDX", "AREACODE", 3, 'C') != SUCCESS)
        return (ERROR);

        /* open database file and accompanying index */
        /* d_SINGLE is non-network mode  */

if (dBopen ("BLAKBOOK.DBF", d_SINGLE, &blakbook) != SUCCESS)
        return (ERROR);
if (dBiopen ("AREACODE.NDX", d_SINGLE, &areacode) != SUCCESS)
        return (ERROR);

        /* put some data into a database and its index */
        /* hardcoded data -- keyboard input is the norm */
        /* 'l' indicates left justified; next arg is (padded) field length */

dBstrcpy (&record[0],  'l', 20, "Joe Blow");                /* name */
dBstrcpy (&record[20], 'l', 25, "369 W. Main St.");          /* address */
dBstrcpy (&record[45], 'l', 25, "Woeis, ME 02233");          /* geography */
if (dBatofld ("555", 3, 0, &record[70]) != SUCCESS)         /* numeric data */
        return (ERROR);                                     /* are different */
if (dBatofld ("7654321", 7, 0, &record[73]) !=SUCCESS)      /* omit hyphen */
        return (ERROR);
dBstrcpy (codekey, 'l', 3, "555");                          /* key: area code */
if (dBakey (areacode, codekey, (RECNUM) 1) !=SUCCESS) /* add the key */
        return (ERROR);

        /* write the record */

if(dBputr (blakbook, (RECNUM) 1, record) != SUCCESS)
        return(ERROR);

        /* close and done */

if (dBclose (blakbook)  != SUCCESS) return (ERROR);
if (dBiclose (areacode) != SUCCESS) return (ERROR);
 }
```

Figure 1 - dBC III Plus

```
* dBase III + code--selects a random record from data base
* the following sequence is coded as a .prg file
* library dct1.exe is assumed to have already been loaded
* load the .bin file
load cfunc
.
* some dBase code goes here
.
* set the return variable
public c_n_result
c_n_result = 0
*optional parameters: here, 1 and 1000 are record number limits
call cfunc with "rand 1, 1000"
* echo random number to screen
? c_n_result
*retrieve a random record
disp all for recno() = c_n_result
.
* more dBase code, etc.
```

Figure 2 - dBase Tools for C

tenance. And anyone who has picked up a computer magazine in the past twelve months is witness to the debate regarding OS/2's future. But it is not this article's purpose to judge the viability of OS/2; rather, it presents libraries as building blocks for whatever you want to do with them.

This essay also does not purport to be a comprehensive rundown of all libraries available for the C language. Rather, it will attempt to be exemplary and note the major categories of applications for which C libraries are available, and indicate the scope and power of each one's uses. At the risk of sounding like an Academy Awards announcer, the categories are:

I. Libraries of linkable object modules from commercial sources, published by a C compiler publisher.

One of the best known examples of this broad category is the dBC III Plus package from Lattice. This package is also available for Amiga C as well as for several major brands of C compilers for the PC. Source code is available for all versions at additional cost. Most of the functions can replace 400-600 lines of C code (minus strings of comments). dBC III Plus can generate database support for LANs, supporting file locking and unlocking, and will also compile under OS/2. There is full dBase III + compatibility; the big advantage is that the product removes the tedium of setting up long header files in one's C code. What is missing are the many callable functions and report/formatting attributes of dBase III +. But there are some things you must do yourself, and the file structure management reduces coding time dramatically.

Although not yet announced, it is inevitable that there will be a dBC IV Plus. For those who don't want the multistation capabilities of dBC III Plus, there is a standard dBC III which corresponds to the former dBase III program and is targeted for single user workstations. Figure 1 outlines a simple yet powerful implementation of dBC III Plus.

II. Libraries published by a software house, targeted exclusively for one or more of its own products.

While on the subject of databases, consider dBase Tools for C and its counterpart, Graphics Tools for C. In case you hadn't noticed, dBase III + is missing a few bells and whistles. Specifically, if you want to perform any higher math or elaborate business computations with dBase III +, you either have to tweak the dBase III + language or perform the computations outside the data base. The dBase Tools for C library takes care of those computations, in line with the dBase III + code. The package includes a cfunct.bin interface module which allows you to call the tools functions from within dBase III +. The latter are contained in dct1.exe, and a bonus is that you can add your own C functions to the basic set provided. To that end, support is included for several popular compilers, with instructions on how to add your own authored code. From there, it is a simple matter of parameter passing with the result returned to a dBase III + variable, just like in C.

Figure 2 shows a simple application. Suppose you want to select a random entry from a data base. You must first set up a random number, and dBase III + does not include anything like random() in the basic package. It is a simple matter of calling the Tools' RAND [w, high] function and obtaining an arbitrary value. This number can then be used to search the data base on recno(). The programmer is, of course, responsible to ascertain that the returned value is within proper ranges of a particular data base.

III. Third-party libraries published by independent commercial groups of authors, targeted for several brands of native compilers.

The degree of cooperation between the publishers of such libraries and their benefactor compiler houses will vary, but in general, the compiler publishers have no arrangements with third-party authors. An exception would be where a company would contract a job out-of-house for a specific applications package). This is another way of saying that such third-party libraries, although they carry the "compatibility" claim along with the appropriate legal exultations and trademark acknowledgements, can be disavowed by the compiler manufacturers. This disclaimer would not apply, however, if a software house were acting as sales agent for the third-party product. Willingness to sell implies willingness to provide technical support for an out-of- house product, but usually only after the compiler publisher has performed a reasonable amount of preflight checkouts. A chain being only as strong as its weakest link, a publish-

```
/* Sequence to clear screen, set a yellow border, print a centered greeting in
   blue.  Only the greeting is in color; on exit, returns to dos prompt.  Border
   remains in set color until another cls is invoked. */

#include <stdio.h>
#include <string.h>
#include <color.h>                    /* mostly definitions */

void cls();                           /* same as dos */
void palette (int, int);              /* gets colors from header file */
void atsay (int, int, char);          /* reminiscent of dBase */
void rcprints (int, int, int, char);  /* no prototypes in header file */

main()
  {
        int line = 0, column = 0;                       /* going to home cursor */
        char *string = "\n\n\n\n";                      /* a few vertical tabs */
        int cols = 80, color = BLUE, page = 0;  /* could set in header file */
        char *message = "Hello, world!";                /* famous first words */

        cls();
        palette (BACKGROUND, YELLOW); /* library function--set border */
        atsay (line, column, string);                   /* tab down a few lines */
        rcprints (cols, color, page, message);          /* output in color; center */
        printf ("\n\n\n");                              /* scoot the prompt lower */
  }
```

Figure 3 - The Greenleaf Functions

er knows that the company's reputation could be soiled by distribution of an inferior third-party product.

As an aside, if technical support for a product is particularly important to you, investigate support policies on a particular package before you take it out of the store. Hardware manufacturers sometimes include brand name software as a package deal with the computer. If any modifications or enhancements have been performed on the basic programs or utilities, the original publisher may refuse support on his product. The reason for this is not indifference, but the fact that the programs may have been furnished to the hardware company as OEM's (original equipment manufacturings). Publishers sell these packages in bulk, in a somewhat generic form, and the purchasing company can do repackaging or even some rewriting of the documentation to conform to its specific hardware attributes. An example of the latter is that of Heath/Zenith, where programs such as WordStar, Word, Multiplan, and the Microsoft Macro Assembler were sold in Heath binders with additions to the manuals that would benefit users of Zenith machines.

Greenleaf Software publishes a very popular set of C libraries of this category, appropriately enough named the Green leaf Functions. This library has been around awhile; there is also a Communications Library and a Data-Windows Library. Here are included all those convenience functions that ANSI C omitted. The functions address DOS, video attributes, the keyboard, the clock, alternate and compound keystrokes, the BIOS, and I/O ports. When's the last time you tried to map logical output to any other than LPT1:? The communications package is a very powerful aid to direct modem control, and is a natural companion to Campbell's C Programmer's Guide to Serial Communications.[3]

Figure 3 shows a good example of Greenleaf filling the gap of often taken for granted functions. CLS to blank the screen and home the cursor in DOS is a sequence most of us type without forethought. Not so easy in C, unless you want to go to the trouble of outputting the appropriate escape sequences each time. Greenleaf makes CLS a callable subroutine. If you're one of those programmers who must know what makes everything "tick", source code is included. The majority of Greenleaf's functions are portable C, but assembly language is used where code must reach into the bowels of the machine.

One other specific example. Remember the old "Zip" program for dBase II? While this utility does not technically constitute a set of library modules, it exemplifies add-on enhancements for a base product that cuts time on tedious activities. "Zip" allowed the user to create input screens by reading specific entries off the video display screen and subsequently generating the @...say...gets that dBase understands. This was one of the very early applications of using terminals (such as the H-19) that could report full screen attributes.

Panel Plus (the code example of figure 4) is a rather elaborate library that lends itself to a wide variety of C-based functions. There are several ways of invoking its powers. Programmers initially experimenting with Panel Plus will probably take advantage of the code generator for creating screen layouts. Herein lies the similarity to Zip (mentioned above). However, the power of this library lies in the willingness of the user to exploit its line-for-line invocation of predigested C routines.

In addition to screen manipulation, Panel Plus will allow creation of pop-up and pull-down menus, elementary graphics displays, and help windows. There is also mouse support available, and Panel Plus will compile under OS/2. Source code is included at no additional cost.

The code segment in figure 4 demonstrates how only a few lines of C code can create a pop-up menu under Panel Plus. It is the programmer's responsibility to set up the information that will appear in the menu lines, to be pointed to in panfcb. From there, the process is one only of passing four parameters (the file pointer, starting line number, highlight type and modifier to highlight a single line) to pamenu() to obtain the menu. A considerable amount of housekeeping is tended by

```
/* pamenu()--subroutine to invoke a pop-up vertical menu. From Panel Plus
   Advanced Screen Manager, (c) 1987 by Roundhill Computer Systems and
   Lattice, Inc.4 */

#include <pnl3.h>          /* includes pfcb.h, which defines
                              field control blocks and named
                              values for panelf function */
panfcb *fp;                /* file pointer; "road map" to
                              locators for menu placement */
int ret;
int start,menht,menhm;     /* start = line number;
                              menht = highlight type;
                              menhm = modifier to highlight a single line */
ret = pamenu(fp,start,menht,menhm);     /* Panel Plus does the rest */
```

Figure 4 - Panel Plus

Panel Plus--the ability to move the highlighted bar with the up/down arrow keys is enabled, and the option of selecting a menu item either from a keystroke of an upper-case letter corresponding to the menu item or pressing the <cr> after having highlighted a choice is made possible. The underlying code of pamenu() also takes care of blanking/restoring the cursor and rewriting the screen upon erasure of the menu.

IV. Windowing and presentation packages.

The most difficult part of writing C code in conjunction with a windows package is to avoid compatibility problems. When windows are introduced, function collisions seem to breed, and code, which heretofore ran bug-free, suddenly exhibits all sorts of bizarre behavior. Herein lies one of the strongest arguments for the old KISS (keep it simple, stupid) maxim, to use the least complicated (read physically smallest) presentation software to avoid the gremlins that seem bound to wreck a well-intended, graphically pleasing program.

One of the most refreshing libraries to come along in some time is the Window Boss. This program (actually a subset of it) has appeared on many bulletin boards across the country. It is usable with several popular brands of C compilers, and as a piece of bulletin board shareware, is complete for small model compiles. What you get for the (current) $60 registration fee is source code and linkable .obj files for the other memory models. You don't get things like mouse and sidebar control, but the product is money well spent for its speed alone. Color is supported, as is multiple windowing.

Figure 5 illustrates a call to a basic Window Boss function in which the screen is painted with a small colored block containing a signon message. As an isolated example it is rather elementary, but the implication that many such calls would make a splendid presentation.

Panel Plus (in III. above) almost fits into this category, but its functionality extends much beyond just windowing. Persons with a flair for nostalgia (and basic loyalty to UNIX) might want to look at Lattice's Curses package. The aforementioned has the advantage of being terminal independent, although not as "pretty" as some of the other programs. Serious windows programmers should already be familiar with the Microsoft Windows Software Development Kit (not to be confused with the basic Windows or Windows 286 program). With its hundreds of callable C routines to implement every one of the Microsoft Windows functions, it is probably the closest approximation to

```
/* Uses wboss library functions to set up a green bordered box near the
   middle of the screen, in which is printed text on a white background with
   black letters. Small memory model is used here. */

#include <windows.h> /* contains duplicates of stdio.h definitions */

main()
  {
        WINDOWPTR w1;           /* window handle */
        int batrib;             /* border attribute */
        int watrib;             /* window attribute */

        batrib = v_setatr (GREEN, WHITE, 0, 0);
        watrib = v_setatr (WHITE, BLACK, 0, 0);

/* Open window at 10, 10 to 30 spaces wide and 3 lines high. wn_open's
   parameters are (page, row, column, width, height, etc.) */

        w1 = wn_open (0, 10, 10, 30, 3, watrib, batrib);
        if (!w1) exit();

/* Print string and wait for key to be struck. Close window on key strike and
   exit. */

        wn_printf (w1, "Hello World!");   /* famous first words again */
        wn_printf (w1, "\nPress any key to continue...");
        v_getch();                        /* wait for a keystroke */
        wn_close (w1);                    /* back to line following
                                             last dos prompt */
}
```

Figure 5 - Window Boss

the OS/2 Presentation Manager functions short of the actual OS/2 version 1.1. Programmers who plan to stay with DOS for some time might want to give the Development Kit a serious look, if for no other reason than to stay abreast of new skills in the presentations genre.

V. Print sources that contain collections of useful routines (or complete thematic programs) suitable for incorporation into larger programs.

Here is where WaldenBooks, B. Dalton Booksellers, WHSmith and all their counterparts begin to look like gold mines. For about the last five years, the proliferation of trade paperback computer science books has increased dramatically. As a result, the prices of books have dropped to the $20-$25 range. The past two years has witnessed a trend for a published work to offer an accompanying diskette, either as part of the book or as a separate order, of the code segments contained therein. The latter can double the shelf price of the book, but for a particularly worthwhile collection of subprograms, the additional cost is well worth the hours of typing saved. There is a particular abundance of literature aimed at Turbo C, Quick C, and UNIX-based C available in trade paperback format at this time.

Some caveats here: many authors (unless the book is a corporate effort) are relating programming experiences which are usually based on only one or two different brands of C compilers, and with only a single version of each one at that. There just isn't the time to purchase and debug a program library under all the popular compilers on the market. Nor do freelance authors have the collective research and development teams at their disposal as do even the medium-sized companies. After all, much of the author's time must be spent at the primary duty--preparing and revising a readable manuscript to allow you to take advantage of such a bargain priced, thick compendium of C libraries!

Regardless of what the preface says, accept the premise that no out-of-book C code will compile flawlessly when entered exactly as typed off the pages, even if the author guarantees compatibilities. The last thing that a technical support representative wants to hear you complain is that "I typed it line-for-line as in the book, and it doesn't work with your compiler." About 100 per cent of the time--guaranteed--the problem is either a typographical error (yours) or a step you overlooked in the author's instructions. So experiment with the code in the book in the spirit in which it was intended--a bargain and a sharing of experiences.

```
/* BOXX.C draws a box given the coordinates of two diagonal corners.
   Copyright (c) 1987 by McGraw-Hill, Inc. Used  by permission.[5]  */

#define TRUE -1
#define FALSE 0
#define ON TRUE
#define OFF FALSE

boxx(x1, y1, x2, y2, style, color, orxor)      /* programmer passes appro- */
int x1, y1, x2, y2, color, orxor;              /* priate screen coordinates */
unsigned style;
{
int xul, yul, xlr, ylr;                        /* the corners */

xul = (x2 x1) ? x1 : x2;                       /* swap coordinates */
yul = (y2 y1) ? y1 : y2;                       /* if necessary */
xlr = (x2 x1) ? x2 : x1;
ylr = (y2 y1) ? y2 : y1;

linef(xul, yul, xlr, yul, color, style, orxor, OFF);
linef(xlr, yul, xlr, ylr, color, style, orxor, OFF);
linef(xlr, ylr, xul, ylr, color, style, orxor, OFF);
linef(xul, ylr, xul, yul, color, style, orxor, OFF);
}
```

Figure 6 - Advanced Graphics in C

One of the better examples of this category is Johnson's Advanced Graphics in C.[6] Figure 6 illustrates one of the many drawing functions in the book; in this case, a subroutine similar to the line [(x1,y1) - (x2,y2)] function in BASIC. Used with the author's GRAPHIQ routines, it is possible to output one or many rectangles to a dot-matrix printer or to the screen. Boxx() calls the linef() macro routine (also detailed in the book), passing the screen coordinates to that function. Orxor and OFF are a couple of housekeeping parameters that take care of line overlap.

The code in this book is available on disk for $21.95, and must be ordered by mail as an option. One of the strengths of this writing is that it offers a lot of detail on accessing EGA attributes through C routines. This is an area missing even from many of the commercially prepared C library packages.

One final note: the serious book collector would do well to investigate the Library of Computer and Information Science, a book club headquartered in Riverside, New Jersey. Numerous C titles appear in their catalog, many of which surface in only the larger bookstores. And LCIS' discounts can be substantial, even for the casual buyer. Also, try to get a copy of the Programmer's Connection Buyer's Guide catalog.[7] Even if you don't purchase anything (and discounts also abound here), keep the catalog around as a comprehensive, annotated reference tool on C libraries. Check out also the public domain catalog of Public Brand Software.[8] Many C libraries are for sale therein at $5 per disk. Several advantages are found here: For one, the catalog tells you when the author expects a gratuity or registration fee (and how much) for the shareware.

```
/* power3.c -- calls library routine 'cube' in small memory model */

#include <stdio.h>
extern int cube(int);

main()
{
        int x;
        printf ("Enter a whole positive number 1 - 15 to be cubed: ");
        scanf ("%d", &x);
        if (x 15 | x 0 )
                printf ("\nEntry out of range; aborting.");
        else
                printf ("\nThe cube of %d is %d.\n", x, cube(x));
}

; cube.asm, which is called as a library function:
;
; subroutine to compute the cube of a passed number
; won't accept bases greater than 15 (to keep code simple)
; called by small memory model
;
 PGROUP GROUP PROG
PROG SEGMENT BYTE PUBLIC 'PROG'
ASSUME  CS:PGROUP               ;caller is responsible for stack
;
        PUBLIC          cube
cube    PROC
        push    bp              ;entry sequence -- save old BP
        mov     bp,sp           ;set stack frame pointer
;
; passed arguments are loaded onto stack
; note that return offset is 4, the length of the pushed
; return address and BP
; maintain integrity of AX--resulting int is returned in these registers
;
        mov ax,[bp+4]   ;load argument into AX
        mov cx,[bp+4]   ;ditto for CX
        mul     cl      ;first multiply (accumulator * CL)
        mul     cl      ;second multiply
;
        pop     bp      ;exit sequence-- restore BP
        ret             ; and return to main program 'cube'
        ENDP            ;caller restores the stack
PROG    ENDS
END
```

Figure 7 - Assembly Language Library Function Called from C

And, there are some unique little programs here--like exotic cross-assemblers--that could well be worth many times the up front $5 tariff.

VI. Libraries callable like C but originating in assembly language.

Admittedly, this article's theme is C libraries. But as one final category it seemed appropriate to look at the opposite process, that of calling assembly language with C and the mechanisms provided by compiler publishers to make the job easier.

Two specific examples here: Microsoft has brought C version 5.1 into synchronization with MASM version 5.1. The process has been simplified to where (at the most elementary level) the .asm code need only specify .MODEL (memory model for C and assembly code) and .CODE (defining the subroutine's bounds in assembler). Other controls are available, as shown in Figure 7, but, if the assembly code is correct, just .MODEL and .CODE will suffice to call an assembly-based subroutine from C. With appropriate declarations from C, parameters are automatically passed to the stack and returned in the AX register. The process is transparent to the programmer. Figure 7 shows the process of calling an assembly language subroutine to cube a number.

Lattice C in its PC version 3.4 release has maintained the convention of handling assembly language calls through the macros dos.mac and lc.mac. There is an 8087.mac which extends code access to the numeric data processor (8087 or 80287 chips). Lattice includes a file called template.asm which provides a framework for assembly language code. Some programmers find this method preferable since it gives a uniform appearance to a collection of individually written assembler subroutines. Other C compilers follow similar conventions in calling assembler (and FORTRAN, COBOL, BASIC, etc.) .obj files; the methods of Microsoft and Lattice are typical. Some compilers offer "in line assembly", allowing the programmer to toggle a switch to signal the beginning and end of assembly code within the C, incorporating the resulting .obj code into a single file. However, from a libraries standpoint, separate compilation and assemblies are more the norm.

Footnotes and Sources

[1] Kernighan, Brian W. and Ritchie, Dennis M. The C Programming Language. Englewood Cliffs, N.J.: Prentice-Hall, 2d ed., 1988, p. 192.

[2] Draft Proposed American National Standard for Information Systems--Programming Language C. Washington, D.C.: The American National Standards Institute, 1/11/88.

[3] Campbell, Joe. C Programmer's Guide to Serial Communications. Indianapolis, Ind.: Howard W. Sams & Company, 1987.

[4] Panel Plus Advanced Screen Manager. Englewood, N.J.: Roundhill Computer Systems Limited (dist. by Lattice, Inc.), 1987, p. 15-15.

[5,6] Johnson, Nelson. Advanced Graphics in C. Berkeley, Cal.: Osborne McGraw-Hill, 1987, pp. 192-193.

[7] The Connection: Programmer's Connection Buyer's Guide. North Canton, Ohio: Programmer's Connection, 1988 (mailing address 7249 Whipple Avenue NW, Canton 44720). Free upon request.

[8] Shareware Catalog and Reference Guide. Indianapolis, Ind.: Public Brand Software, 1988 (mailing address P.O. Box 51315, Indianapolis 46251). $1.00 a copy, but new buyers usually receive several complimentary copies once on PBS' mailing list.

List of Products and Resources Mentioned (in Order of Appearance)

dBC III ($250) and dBC III Plus ($750)
(for Library Source, add $250 and $750 respectively)
Curses ($125)
(for Library Source add $250)
Panel Plus v.1.10 ($495); UNIX/XENIX version ($795)

Lattice, Incorporated
2500 South Highland Avenue
Lombard, IL 60148

dBase Tools for C ($89)
Graphics Tools for C ($89)

Ashton-Tate Publishing Group
20101 Hamilton Avenue
Torrance, CA 90502-1319

The Greenleaf Functions v.3.0 ($185)
The Greenleaf Communications Library v.2.0 ($185)
Greenleaf DataWindows ($225)

Greenleaf Software
16479 Dallas Parkway, Suite 570
Dallas, TX 75243

The Window Boss ($60)

Star Guidance Consulting
273 Windy Drive
Waterbury, Ct 06705

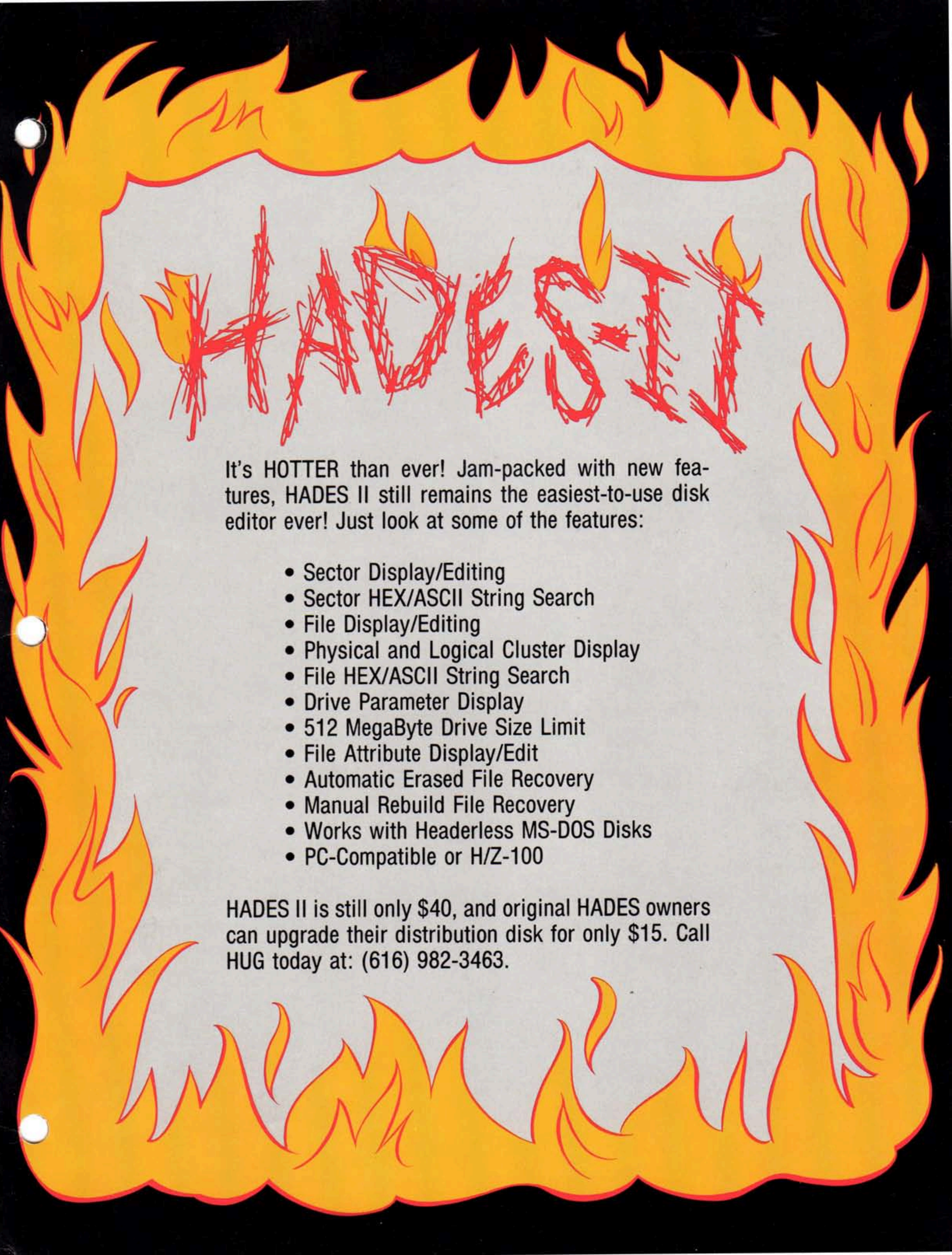
Microsoft Windows Software Development Kit ($500)

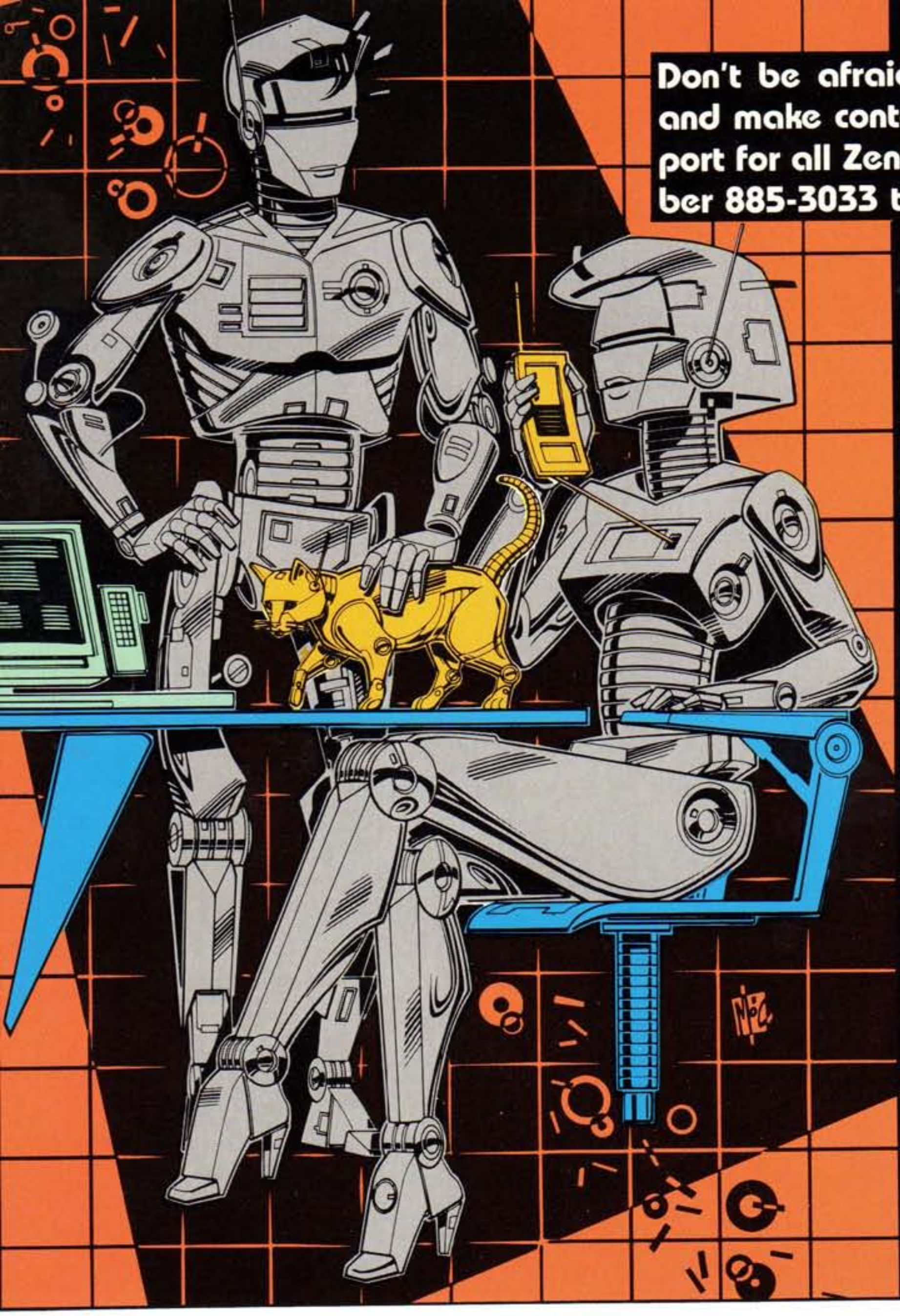
Microsoft Corporation
16011 N.E. 36th Way
Redmond, WA 98073

The Library of Computer and Information Sciences

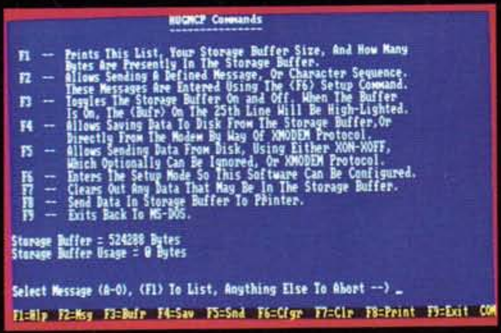
LCIS
Membership Department
Riverside, NJ 08075-9889

✱

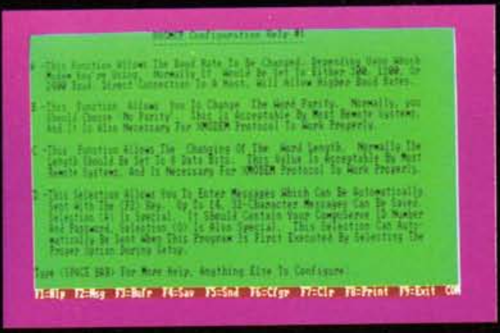




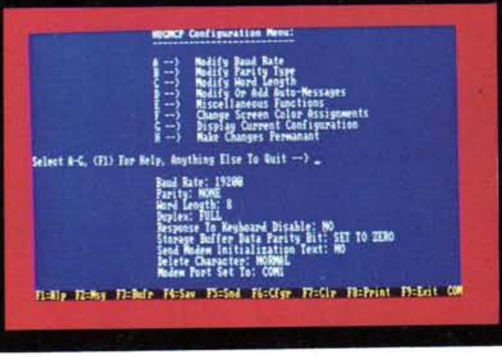
Don't be afraid to communicate! Get HUGMCP and make contact the easy way. Now with support for all Zenith Laptops, order HUG Part number 885-3033 today.



```
                  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 (Bufr) On The 25th Line Will Be High-Lighted.
F4 --  Allows Saving Data To Disk From The Storage Buffer,Or
       Directly From The Modem 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 PPinter.
F9 --  Exits Back To MS-DOS.

Storage Buffer = 524288 Bytes
Storage Buffer Usage = 0 Bytes

Select Message (A-O), (F1) To List, Anything Else To Abort --> _

F1=Hlp  F2=Msg  F3=Bufr  F4=Sav  F5=Snd  F6=Cfgr  F7=Clr  F8=Print  F9=Exit  COM
```



```
                 HUGMCP Configuration Help #1

A -This Function Allows The Baud Rate To Be Changed. Depending Upon Which
   Modem You're Using, Normally It Would Be Set To Either 300, 1200, Or
   2400 Baud. Direct Connection To A Host, Will Allow Higher Baud Rates.

B -This Function Allows You To Change The Word Parity. Normally, you
   Should Choose 'No Parity'. This Is Acceptable By Most Remote Systems,
   And It Is Also Necessary For XMODEM Protocol To Work Properly.

C -This Function Allows The Changing Of The Word Length. Normally The
   Length Should Be Set To 8 Data Bits. This Value Is Acceptable By Most
   Remote Systems, And Is Necessary For XMODEM Protocol To Work Properly.

D -This Selection Allows You To Enter Messages Which Can Be Automatically
   Sent With The (F2) Key. Up To 14, 32-Character Messages Can Be Saved.
   Selection (A) Is Special. It Should Contain Your CompuServe ID Number
   And Password. Selection (O) Is Also Special. This Selection Can Auto-
   matically Be Sent When This Program Is First Executed By Selecting The
   Proper Option During Setup.

Type (SPACE BAR) For More Help, Anything Else To Configure:

F1=Hlp  F2=Msg  F3=Bufr  F4=Sav  F5=Snd  F6=Cfgr  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 Permanant

Select A-G, (F1) For Help, Anything Else To Quit --> _

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

F1=Hlp  F2=Msg  F3=Bufr  F4=Sav  F5=Snd  F6=Cfgr  F7=Clr  F8=Print  F9=Exit  COM
```

Heath Users' Group / ZENITH

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

POSTMASTER: If undeliverable, please do not return.

$2.50
P/N 885-2114