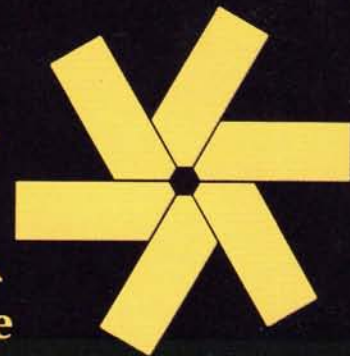


REMark

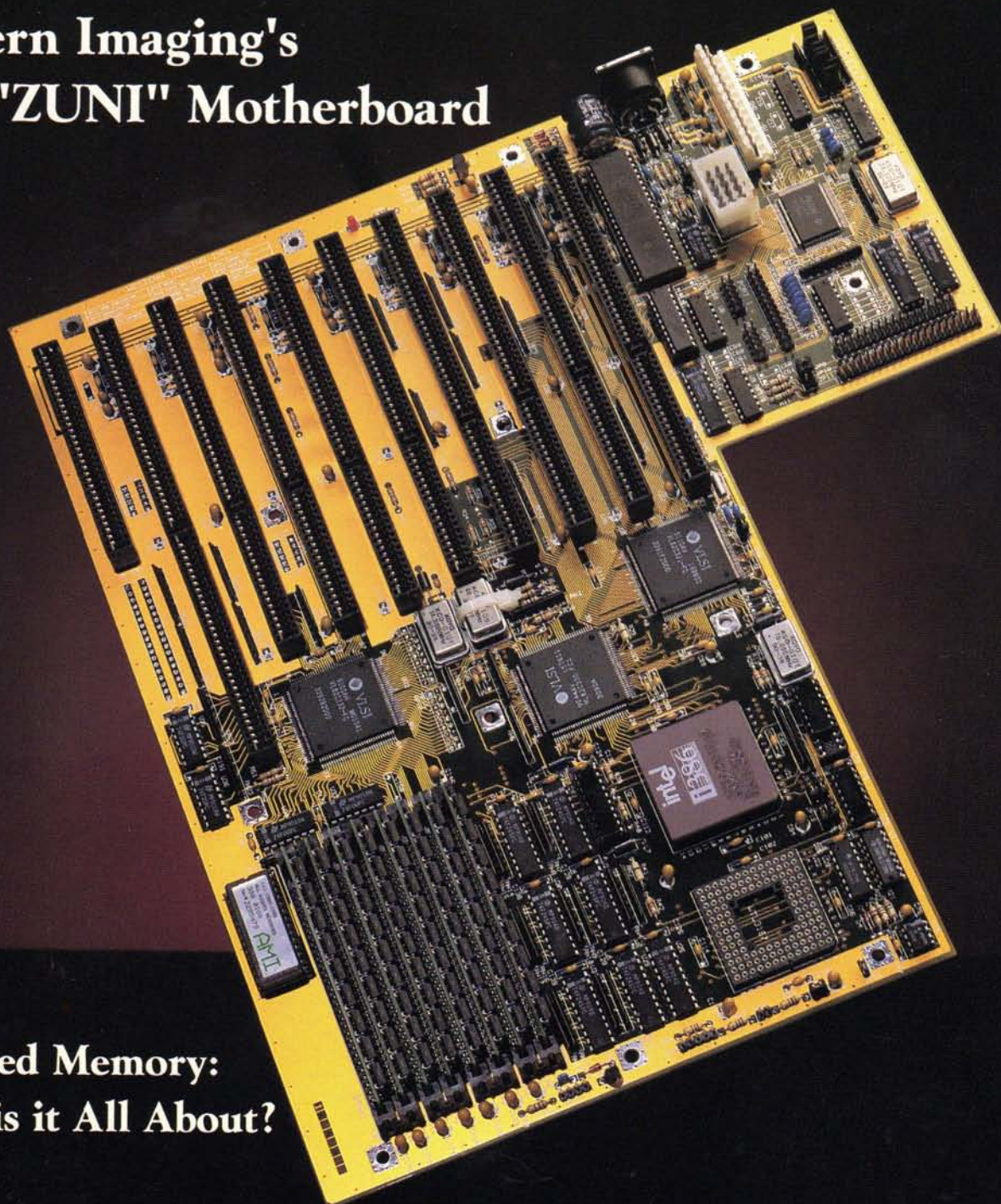
October 1991



The Official Zenith Data Systems Users Magazine

Western Imaging's Z-33 "ZUNI" Motherboard

Page 5



Expanded Memory: "What is it All About?"

Page 27

Small Computer System Interface (SCSI)

Page 9



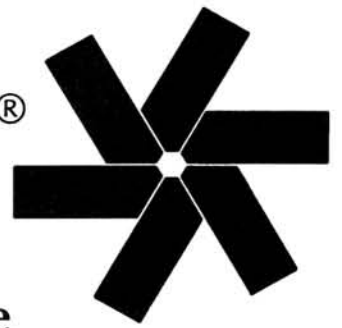
Share the Knowledge!

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 product, be it software or hardware, for your system? By submitting this information in the form of a major article, you can share with others the knowledge of a particular subject. REMark magazine is currently looking for authors (novice or professional) to write articles. Even if you have never written before, give it a try! As a REMark author, you will receive up to \$400 for each article accepted and published. (For more information on current policies, call Lori Lerch at 616-982-3794.) So, Let's get to it and ...

Share the Knowledge!

REMark[®]

October 1991



The Official Zenith Data Systems Users Magazine

Western Imaging's Z-33 "ZUNI" Motherboard

Jim Fonte 5

Choosing a Computer Monitor

David R. Veit 7

Small Computer System Interface

Nathan Baker 9

All Around the eaZy PC

Laszlo M. Vesei 15

On the Leading Edge

William M. Adney 19

External Portable Hard Drive for Your Laptop or PC-Compatible

Henry E. Fale 25

Expanded Memory: What is it All About?

John Eichenberger 27

The World of WP50 and Its Wonders Part VI

Salli Brackett 33

The Hewlett-Packard LaserJet IIP

William Wills 35

Getting the Most From Your Computer Part 5

John Lewis 38

Shrinking WordPerfect 5.1 to Fit on a MinisPort ZL-2

Jerry Klinke 43

Modems — Part 5

Robert C. Brenner 45

Advertising

	Page No.
FBE Research Co., Inc.	24
LS Software	48
QuikData, Inc.	14
WS Electronics	26

Resources

Software Price List	2
Renewal Form	4

Managing Editor
Jim Buszkiewicz
(616) 982-3837

Software Engineer
Pat Swayne
(616) 982-3463

Production Coordinator
Lori Lerch
(616) 982-3794

Secretary
Lisa Cobb
(616) 982-3463

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

ZUG
Software Orders
(616) 982-3463

Contributing Editor
William M. Adney

Printer
Imperial Printing
St. Joseph, MI

Contributing Editor
Robert C. Brenner

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

To Locate your Nearest:

Dealer 1-800-523-9393
Service Center 1-800-777-4630

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

* U.S. Funds

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

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

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

Articles submitted by users and published in REMark, which describe hardware modifications, are not supported by Zenith Data Systems Computer Centers.

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

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

Copyright (c) 1991, Zenith Users' Group

Software

PRODUCT NAME	PART NUMBER	OPERATING	DESCRIPTION	PRICE
		SYSTEM		
H8 - H/Z-89/90				
ACTION GAMES	885-1220-[37]	CPM	GAME	20.00
ADVENTURE	885-1010	HDOS	GAME	10.00
ASCIRITY	885-1238-[37]	CPM	AMATEUR RADIO	20.00
AUTOFILE (Z80 ONLY)	885-1110	HDOS	DBMS	30.00
BHBASIC SUPPORT PKG	885-1119-[37]	HDOS	UTILITY	20.00
CASTLE	885-8032-[37]	HDOS	ENTERTAINMENT	20.00
CHEAPCALC	885-1131-[37]	HDOS	SPREADSHEET	20.00
CHECKOFF	885-8010	HDOS	CHKBK SOFTWARE	25.00
DEVICE DRIVERS	885-1105	HDOS	UTILITY	20.00
DISK UTILITIES	885-1213-[37]	CPM	UTILITY	20.00
DUNGEONS & DRAGONS	885-1093-[37]	HDOS	GAME	20.00
FLOATING POINT PKG	885-1063	HDOS	UTILITY	18.00
GALACTIC WARRIORS	885-8009-[37]	HDOS	GAME	20.00
GALACTIC WARRIORS	885-8009-[37]	CPM	GAME	20.00
GAMES 1	885-1029-[37]	HDOS	GAMES	18.00
HARD SECT SUPPORT PKG	885-1121	HDOS	UTILITY	30.00
HDOS PROG. HELPER	885-8017	HDOS	UTILITY	16.00
HOME FINANCE	885-1070	HDOS	BUSINESS	18.00
HUG DISK DUP UTILITY	885-1217-[37]	CPM	UTILITY	20.00
HUG SOFTWARE CATALOG	885-4500	VARIOUS	PROD TO 1982	9.75
HUGMAN & MOVIE ANIM	885-1124	HDOS	ENTERTAINMENT	20.00
INFO SYS AND TEL. & MAIL SYS	885-1108-[37]	HDOS	DBMS	30.00
LOGBOOK	885-1107-[37]	HDOS	AMATEUR RADIO	30.00
MAGBASE	885-1249-[37]	CPM	MAGAZINE DB	25.00
MISCELLANEOUS UTILITIES	885-1089-[37]	HDOS	UTILITY	20.00
MORSE CODE TRANSCEIVER	885-8016	HDOS	AMATEUR RADIO	20.00
MORSE CODE TRANSCEIVER	885-8031-[37]	CPM	AMATEUR RADIO	20.00
PAGE EDITOR	885-1079-[37]	HDOS	UTILITY	25.00
PROGRAMS FOR PRINTERS	885-1082	HDOS	UTILITY	20.00
REMARK VOL 1 ISSUES 1-13	885-4001	N/A	1978 TO DEC '80	20.00
RUNOFF	885-1025	HDOS	TEXT PROC	35.00
SCICALC	885-8027	HDOS	UTILITY	20.00
SMALL BUSINESS PACKAGE	885-1071-[37]	HDOS	BUSINESS	75.00
SMALL-C COMPILER	885-1134	HDOS	LANGUAGE	30.00
SOFT SECTOR SUPPORT PKG	885-1127-[37]	HDOS	UTILITY	20.00
STUDENT'S STATISTICS PKG	885-8021	HDOS	EDUCATION	20.00
SUBMIT (Z80 ONLY)	885-8006	HDOS	UTILITY	20.00
TERM & HTOC	885-1207-[37]	CPM	COMMUN & UTIL	20.00
TINY BASIC COMPILER	885-1132-[37]	HDOS	LANGUAGE	25.00
TINY PASCAL	885-1086-[37]	HDOS	LANGUAGE	20.00
UDUMP	885-8004	HDOS	UTILITY	35.00
UTILITIES	885-1212-[37]	CPM	UTILITY	20.00
UTILITIES BY PS	885-1126	HDOS	UTILITY	20.00
VARIETY PACKAGE	885-1135-[37]	HDOS	UTILITY & GAMES	20.00
WHEW UTILITIES	885-1120-[37]	HDOS	UTILITY	20.00
XMET ROBOT X-ASSEMBLER	885-1229-[37]	CPM	UTILITY	20.00
Z80 ASSEMBLER	885-1078-[37]	HDOS	UTILITY	25.00
Z80 DEBUGGING TOOL (ALDT)	885-1116	HDOS	UTILITY	20.00
H8 - H/Z-89/90 - H/Z-100 (Not PC)				
ADVENTURE	885-1222-[37]	CPM	GAME	10.00
BASIC-E	885-1215-[37]	CPM	LANGUAGE	20.00
CASSINO GAMES	885-1227-[37]	CPM	GAME	20.00
CHEAPCALC	885-1233-[37]	CPM	SPREADSHEET	20.00
CHECKOFF	885-8011-[37]	CPM	CHKBK SOFTWARE	25.00
COPYDOS	885-1235-[37]	CPM	UTILITY	20.00
DISK DUMP & EDIT UTILITY	885-1225-[37]	CPM	UTILITY	30.00
DUNGEONS & DRAGONS	885-1209-[37]	CPM	GAMES	20.00
FAST ACTION GAMES	885-1228-[37]	CPM	GAME	20.00
FUN DISK I	885-1236-[37]	CPM	GAMES	20.00
FUN DISK II	885-1248-[37]	CPM	GAMES	35.00
GAMES DISK	885-1206-[37]	CPM	GAMES	20.00
GRADE	885-8036-[37]	CPM	GRADE BOOK	20.00
HRUN	885-1223-[37]	CPM	HDOS EMULATOR	40.00
HUG FILE MANAGER & UTILITIES	885-1246-[37]	CPM	UTILITY	20.00
HUG SOFTWARE CAT UPDT #1	885-4501	VARIOUS	PROD 1983 TO 1985	9.75
KEYMAP CPM-80	885-1230-[37]	CPM	UTILITY	20.00
MBASIC PAYROLL	885-1218-[37]	CPM	BUSINESS	60.00
NAVPROGSEVEN	885-1219-[37]	CPM	FLIGHT UTILITY	20.00
SEA BATTLE	885-1211-[37]	CPM	GAME	20.00
UTILITIES BY PS	885-1226-[37]	CPM	UTILITY	20.00
UTILITIES	885-1237-[37]	CPM	UTILITY	20.00
X-REFERENCE UTIL FOR MBASIC	885-1231-[37]	CPM	UTILITY	20.00
ZTERM	885-3003-[37]	CPM	COMMUNICATIONS	20.00

Price List

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

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
H/Z-100 (Not PC) Only				
CARDCAT	885-3021-37	MSDOS	BUSINESS	20.00
CHEAPCALC	885-3006-37	MSDOS	UTILITY	20.00
CHECKBOOK MANAGER	885-3013-37	MSDOS	BUSINESS	20.00
CP/EMULATOR	885-3007-37	MSDOS	CPM EMULATOR	20.00
DBZ	885-8034-37	MSDOS	DBMS	25.00
DUNGN & DRAGONS (ZBASIC)	885-3009-37	MSDOS	GAME	20.00
ETCHDUMP	885-3005-37	MSDOS	UTILITY	20.00
EZPLOT II	885-3049-37	MSDOS	PRINTER PLOT UTIL	25.00
GAMES (ZBASIC)	885-3011-37	MSDOS	GAMES	20.00
GAMES CONTEST PACKAGE	885-3017-37	MSDOS	GAMES	25.00
GAMES PACKAGE II	885-3044-37	MSDOS	GAMES	25.00
GRAPHIC GAMES (ZBASIC)	885-3004-37	MSDOS	GAMES	20.00
GRAPHICS	885-3031-37	MSDOS	UTILITY	20.00
HELPSCREEN	885-3039-37	MSDOS	UTILITY	20.00
HUG BKGDR PRINT SPOOLER	885-1247-37	CPM	UTILITY	20.00
KEYMAC	885-3046-37	MSDOS	UTILITY	20.00
KEYMAP	885-3010-37	MSDOS	UTILITY	20.00
KEYMAP CPM-85	885-1245-37	CPM	UTILITY	20.00
MATHFLASH	885-8030-37	MSDOS	EDUCATION	20.00
ORBITS	885-8041-37	MSDOS	EDUCATION	25.00
POKER PARTY	885-8042-37	MSDOS	ENTERTAINMENT	20.00
SCICALC	885-8028-37	MSDOS	UTILITY	20.00
SKYVIEWS	885-3015-37	MSDOS	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 PKG	885-3018-37	MSDOS	VARIOUS SPRDST	25.00
TREE-ID	885-3036-37	MSDOS	TREE IDENTIFIER	20.00
USEFUL PROGRAMS I	885-3022-37	MSDOS	UTILITIES	30.00
UTILITIES	885-3008-37	MSDOS	UTILITY	20.00
ZPC II	885-3037-37	MSDOS	PC EMULATOR	60.00
ZPC UPGRADE DISK	885-3042-37	MSDOS	UTILITY	20.00
H/Z-100 and PC Compatibles				
ADVENTURE	885-3016	MSDOS	GAME	10.00
BACKGRD PRINT SPOOLER	885-3029	MSDOS	UTILITY	20.00
BOTH SIDES PRINTER UTILITY	885-3048	MSDOS	UTILITY	20.00
CXREF	885-3051	MSDOS	UTILITY	17.00
DEBUG SUPPORT UTILITIES	885-3038	MSDOS	UTILITY	20.00
DPATH	885-8039	MSDOS	UTILITY	20.00
HADES II	885-3040	MSDOS	UTILITY	40.00
HEPCAT	885-3045	MSDOS	UTILITY	35.00
HUG EDITOR	885-3012	MSDOS	TEXT PROCESSOR	20.00
HUG MENU SYSTEM	885-3020	MSDOS	UTILITY	20.00
HUG SOFTWARE CAT UPD #1	885-4501	MSDOS	PROD 1983 - 1985	9.75
HUGMCP	885-3033	MSDOS	COMMUNICATION	40.00
ICT 8080 - 8088 TRANSLATOR	885-3024	MSDOS	UTILITY	20.00
MAGBASE	885-3050	VARIOUS	MAG DATABASE	25.00
MATT	885-8045	MSDOS	MATRIX UTILITY	20.00
MISCELLANEOUS UTILITIES	885-3025	MSDOS	UTILITIES	20.00
PS' PC & Z100 UTILITIES	885-3052	MSDOS	UTILITIES	20.00
REMARK VOL 8 ISSUES 84-95	885-4008	N/A	1987	25.00
REMARK VOL 9 ISSUES 96-107	885-4009	N/A	1988	25.00
REMARK VOL 10 ISSUES 108-119	885-4010	N/A	1989	25.00
REMARK VOL 11 ISSUES 120-131	885-4011	N/A	1990	25.00
SCREEN DUMP	885-3043	MSDOS	UTILITY	30.00
UTILITIES II	885-3014	MSDOS	UTILITY	20.00
Z100 WORDSTAR CONNECTION	885-3047	MSDOS	UTILITY	20.00
PC Compatibles				
CARDCAT	885-6006	MSDOS	CAT SYSTEM	20.00
CHEAPCALC	885-6004	MSDOS	SPREADSHEET	20.00
CLAVIER	885-6016	MSDOS	ENTERTAINMENT	20.00
CP/EMULATOR II & ZEMULATOR	885-6002	MSDOS	CPM & Z100 EMUL	20.00
DUNGEONS & DRAGONS	885-6007	MSDOS	GAME	20.00
EZPLOT II	885-6013	MSDOS	PRINTER PLOT UTIL	25.00
GRADE	885-8037	MSDOS	GRADE BOOK	20.00
HAM HELP	885-6010	MSDOS	AMATEUR RADIO	20.00
KEYMAP	885-6001	MSDOS	UTILITY	20.00
LAPTOP UTILITIES	885-6014	MSDOS	UTILITIES	20.00
PS' PC UTILITIES	885-6011	MSDOS	UTILITIES	20.00
POWERING UP	885-4604	N/A	GUIDE TO USING PCs	12.00
SCREEN SAVER PLUS	885-6009	MSDOS	UTILITIES	20.00
SKYVIEWS	885-6005	MSDOS	ASTRONOMY UTIL	20.00
TCSPELL	885-8044	MSDOS	SPELLING CHECKER	20.00
ULTRA RTTY	885-6012	MSDOS	AMATEUR RADIO	20.00
YAUD (YET ANOTHER UTIL DSK)	885-6015	MSDOS	UTILITIES	20.00

Attention!!

Zenith Data Systems Owners

When ordering ZUG software, be sure to specify what disk format you would like us to put it on. If you have an H-8, H/Z-89, or H/Z-90, you have the choice of using hard- or soft-sectored disks depending on your drive type. Order soft-sectored by adding a -37 to the end of the part number (i.e., 885-8009-37). Leaving off the -37 specifies a hard-sectored disk (i.e., 885-8009). If you own an H/Z-100 (not PC) series computer, you will always use the -37 at the end of the part number. For PC users, you have the choice of 5-1/4" (-37), 3.5" (-80), or 2" (-90) disks. Just add this number to the end of the ZUG part number (i.e., 885-3009-37, 885-3007-80, 885-3007-90).

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

ORDERING INFORMATION

For VISA, MasterCard, and American Express phone orders, telephone the Zenith Users' Group directly at (616) 982-3463. Have the part number(s), description(s), and quantity ready for quick processing. By mail, send your order, plus 10% postage and handling (\$1.00 minimum charge, up to a maximum of \$5.00) to: Zenith Users' Group, P.O. Box 217, Benton Harbor, MI 49023-0217. VISA, MasterCard and American Express require minimum \$10.00 order. No C.O.D.s accepted.

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



REMark Magazine Subscription & ZLink/COM1 Bulletin Board Information

Your subscription entitles you to receive REMark, our monthly magazine containing articles specific to Zenith Data Systems computer and generally to other PC Compatible computers. All articles in REMark are submitted by readers like you. We welcome YOUR articles, and will pay you for any we accept!

A REMark subscription also allows you full access to the ZLink-COM1 bulletin board system (COM1, for short). There are many, many megabytes of free and shareware software available for downloading to registered COM1 users. Full access also lets you order products from the "Bargain Centre" section of COM1. The money you can save in the Keyboard Shopping Club will pay for decades of REMark subscriptions.


Last, but definitely not least, your subscription puts you in touch with thousands of other Zenith Data System computer users, from whom invaluable information can be exchanged.

REMark subscriptions, currently \$22.95, can be obtained in one of three ways. First, by ordering one on the COM1 bulletin board (see the Keyboard Shopping Club section); second, by phone with VISA, MasterCard, or American Express; and third, through the US Mail using a credit card, money order or check made payable to: Zenith Data Systems. Our address is:

Zenith Data Systems Users' Group
P.O. Box 217
Benton Harbor, MI 49023-0217
(616) 982-3463

Once you receive your ID number, registration on the COM1 BBS is NOT automatic. It requires that you log on, enter your first name and last name EXACTLY as they appear on your REMark mailing label, and then enter your ID number as your password. The FIRST time you access the board, you must elect to start a NEW ACCOUNT and answer the various questions. Once you've done this, our automated scanner will compare the system's database against the subscription database. If you made no mistakes, you will be verified and given full access within 24 hours.

Once you've been authorized as a full member, several important things happen. First, you're given full downloading privileges of up to one megabyte per day. Secondly, you'll have full access to the message boards. And finally, you'll be able to take full advantage of the Bargain Centre product savings.

 *Detach this form, enclose your check, money order or credit card information (no cash please).*

REMark Subscription / Renewal Form

New Member: Yes No Credit Card # _____

ID Number: _____ Exp. Date _____

Address Change?

		Renew	New
Name: _____	U.S. Bulk Mail	<input type="checkbox"/> 19.95	<input type="checkbox"/> 22.95
Address: _____	U.S. First Class	<input type="checkbox"/> 32.95	<input type="checkbox"/> 37.95
City, State, Zip: _____	APO/FPO Surface Overseas	<input type="checkbox"/> 32.95	<input type="checkbox"/> 37.95
Daytime Phone #: () _____	Air Printed Overseas	<input type="checkbox"/> 52.95	<input type="checkbox"/> 57.95

Western Imaging's Z-33 "ZUNI" Motherboard

Jim Fonte, KK9T
712 Thunderbird Drive
P.O. Box 316
New Carlisle, IN 46552

Introduction

As an amateur radio (HAM) operator involved in OSCAR (Orbital Satellite Carrying Amateur Radio) operation, I have the need to use a computer in the ham shack for a variety of tasks. These chores include:

1. Tracking the OSCAR satellites.
2. Controlling my OSCAR station antennas' azimuth and elevation as a function of the tracking program.
3. Automatically correcting the transmitter frequency to compensate for change in frequency that occurs when the signal is relayed by the satellite (Doppler shift).
4. Logging of the radio contacts.

Originally, it was suggested that I use the Zenith Data Systems Z-248, but it soon became obvious that I needed a much faster computer system to handle my requirements . . . *WITHOUT SPENDING A FORTUNE!*

Enter the Western Imaging, Inc. Z-33 "ZUNI" motherboard... This INTEL 80386DX-33-based upgrade is designed to replace the Z-248's motherboard and most of the plug-in cards. The modification is aimed at making your Zenith Data Systems Z-248 system into a true 32-bit '386 machine at a modest cost. By reusing the video and the floppy/hard disk controller boards on the new motherboard, one can now experience the speed and power of the '386 processor without having to "ash-can" your existing system!

Some of the listed features of the Z-33 "ZUNI" are:

- Direct replacement of the Z-248 motherboard.
- 10 expansion slots (eight 16-bit, one 8-bit, one 8/32-bit with direct access to the internal bus).
- VLSI Topcat chip set.
- M5105 "super" I/O chip — two serial ports, one parallel port, a floppy controller capable of handling two floppies and

an IDE hard disk.

- Bus speed is controllable at 33, 15, 10, or 8 MHz.
- Will support up to 64 MB of RAM.
- Supports the Weitek 3167 (Weitek "specific" applications) and/or the INTEL 80387 co-processors.
- User programmable memory page mode, 2- or 4-way block or

Engineering at Western Imaging, "This is the fastest, non-CPU cacheing '386/33 MHz upgrade motherboard on the market...!"

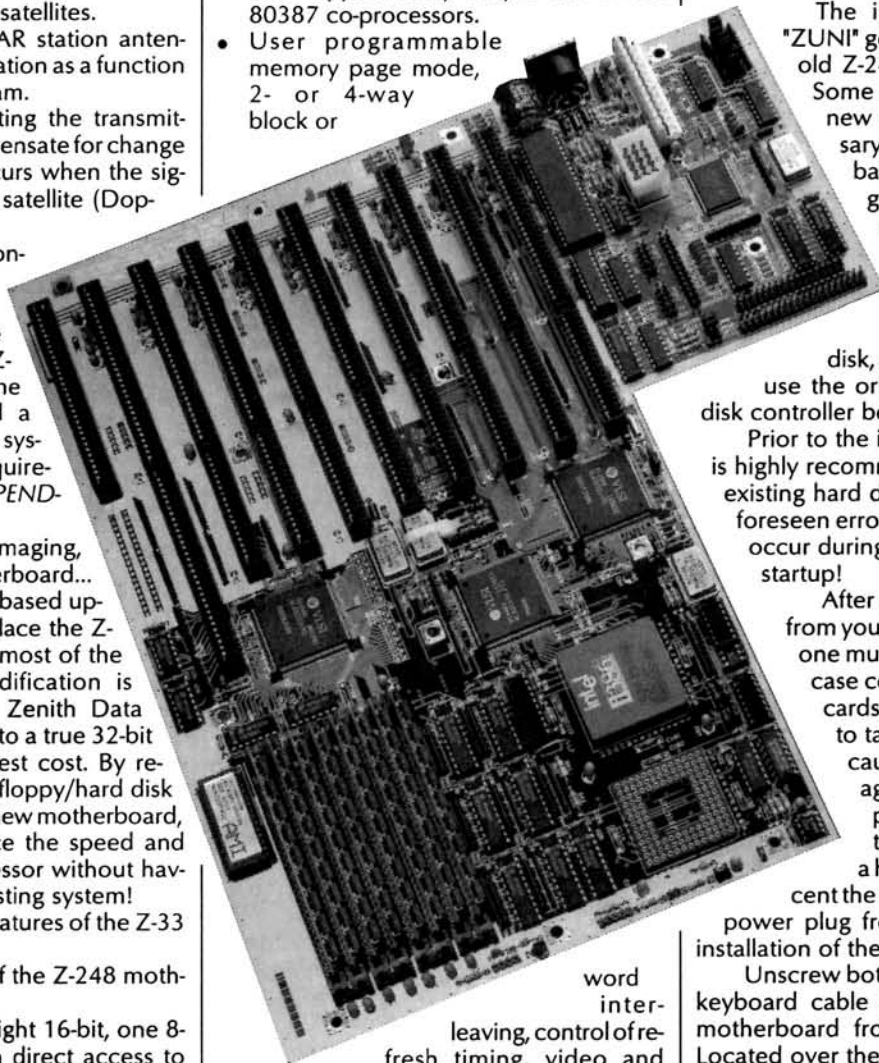
Installation

The installation of the Z-33 "ZUNI" goes very quickly once the old Z-248 "stuffing" is removed. Some strapping options on the new motherboard are necessary to configure the backup battery to service, power good, turbo mode, serial and parallel ports, and whether the user is going to use the on-board 5105 I/O controller for the floppies and the hard disk, or if the user is going to reuse the original Z-248 floppy/hard disk controller board.

Prior to the installation of the Z-33, it is highly recommended to back up your existing hard disk files to avoid any unforeseen errors or problems that might occur during the new motherboards' startup!

After removing the AC power from your present Z-248 machine, one must remove the mainframe case cover and all of the plug-in cards, remembering of course, to take proper anti-static precautions to avoid chip damage. Next, unplug the molex power supply cable from the Z-248 motherboard. If a hard disk is mounted adjacent the motherboard, unplug the power plug from it to facilitate easier installation of the new motherboard.

Unscrew both the ground strap of the keyboard cable socket and the original motherboard from its frame mounting. Located over the rear screw stakes of the motherboard mount are a plastic spacer and a long grounding plate. It is absolutely important that these two items be removed to eliminate the possibility of interference and damage to the new motherboard and/



- word inter-leaving, control of refresh timing, video and BIOS shadowing, I/O timing, DMA timing, and more.
- Compatible with all software based on 100% ISA standard architecture.
- According to Mr. Dave Krauss, VP of

Clock:	DMA Clock = CLK/2 Non-Turbo Mode = CLK/4 Bus Mode = Synchronous Synchronous Bus Clock from TCLK = /3
Wait States:	ROM = 1 16-Bit DMA = 1 8-Bit I/O = 0 16-Bit I/O = 0
Memory Options:	Bank A = Page Mode Bank B = Page Mode Interleave Bank A = Block Interleave Bank B = Block
Extended Memory Options:	Global EMS = No EMS Backfill = No
Refresh Rate:	On-Board Memory = /2 Slot Memory = /2
CPU & DRAM Configuration:	CPU-33/80 ns. DRAM

Figure 1

or the plug-in boards.

Again, using proper anti-static precautions, place the rear of the Z-33 "ZUNI" board into the rear of the mainframe at a downward angle and gently lower the front of the board into position. Use the supplied screws for mounting, but DO NOT OVERTIGHTEN! Make sure the board is level on the mounting stakes and positioned without any mechanical stress before tightening down the mounting screws.

Replace the molex power connector to the power socket of the new motherboard and the power connector to the rear of the hard disk.

Install the supplied serial and parallel port connectors' mounting plate(s) as directed by the manufacturer. Find a convenient location for the video card the floppy/hard disk controller to reside on the new motherboard and install.

Now carefully check all of the suggested strapping options for your particular applications once more for accuracy, replace the mainframe cover, plug in the video cable, and finally, the AC power cord.

Follow the manufacturers' startup/setup procedures carefully. A detailed manual is supplied to cover every parameter of system configuration. Should questions arise, Western Imaging technical support is available to assist.

My particular Z-33 "ZUNI" motherboard 'XCMOS' values are set in Figure 1.

After the new system is configured and running, it is suggested that hard drive reformatting and operating system re-installation be controlled and accomplished by the new Z-33 system to avoid any possible read or write "arguments" that may occur with a change of the controlling CPU and/or system access speeds.

With regards to this upgrade... WOW! What a difference a new '386/33 processor and more flexible BIOS makes. I have used DOS 3.3+, 4.01, OS/2, and Digital

Research DOS 5.0 with this new system and all seem to work just fine (with some exceptions re: V4.01). I also had the opportunity to become absolutely frustrated with Norton Utilities V4.5 and V6.0 with regard to V4.5 "SI" and V6.0 Benchmarks. Both of the Norton series of tests vary greatly as a function of how the CONFIG.SYS file is loaded. An example being, if the CONFIG.SYS does not build any VDISK or use any type of extended memory manager (HIMEM or SMART.DRV, etc.), you'll get one set of results. Modify the CONFIG.SYS file to include any of the above and you'll see the results. The Norton Benchmarks can be more than a little deceiving if you're not paying attention. If you pay attention to what is loaded in the CONFIG.SYS file and how it affects your systems memory management and allocations, you can show the Norton speed tests at just about any speed (including a speed that is slower than what you want!).

The Byte Magazine Sieve Benchmark V2.1, however, did give results that I would be more likely to believe and trust. Figure 2 shows some interesting comparisons:

As you can see, the operating system can have a lot to do with how efficient the computer is actually "crunching the numbers"! I have become absolutely enraptured by the Digital Research DOS 5.0 used with the Z-33. DR DOS really lets you throttle up the Z-33 by utilizing its

Computer	DOS	Sieve Test Result
Z-248	3.3+	6.8X (Faster then XT)
Zenith Data Systems '386/33 (CPU Cacheing)	4.01	29.1X
Western Imaging Z-33 "ZUNI" (Non-Cacheing)	3.3+	20.7X
	4.01	20.5X
	OS/2	23.5X
	DR DOS 5.0	23.7X

Figure 2

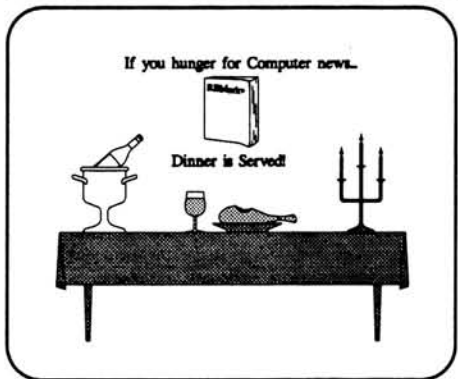
CONFIG.SYS file in such a way as to selectively configure it upon startup to avoid any expanded memory manager conflicts that will occur if one uses a variety of different software (Windows, DesqView, Software Carousel, etc.) at different times.

The addition of a numeric co-processor also enhances the Z-33 very nicely. You'll see the speed difference when using Quattro Pro or any other number crunching spreadsheets. The actual completion time of a series of calculations being reduced to about one-fifth on my machine. I am using the 80387 clone made by ULSI, and really don't see much difference between it and the INTEL chip. I am told that the CYRIX co-processor is even faster, but obviously there may be a point of diminishing returns... price may be the factor.

All in all, the Western Imaging, Inc. Z-33 motherboard upgrade for the Zenith Data Systems Z-248 machine is well worth the investment, in my opinion. My ham shack computer system can now multi-task the satellite tracking program, the logging program, and the antenna positioning program simultaneously, and it still could do more (if I could find a few more hands and fingers!).

Western Imaging, Inc. is offering the Z-33 motherboard with or without extended memory to suite the user's needs. Contact Ms. Tammy Robinson, Account Executive at Western Imaging, Inc. for more detailed information and special pricing. The address is:

Western Imaging, Inc.
115 Constitution Drive
Menlo Park, CA 94025
Phone (415) 325-0300
FAX (415) 325-0499





Choosing a Computer Monitor

David R. Veit
Technical Writer
Zenith Data Systems

With so many types of video display monitors and video subsystems available to PC users these days, it is no wonder that many users simply don't know where to turn when an intelligent decision must be made concerning their choice of a computer display. To make choices a little easier, I will try to remove some of the mystery and mystique surrounding computer monitors.

Before choosing any particular model over another, it helps to know a little about what types are available, how they operate, and most importantly, you as the user must be keenly aware of how you want to utilize the monitor. Know your application! Many users think "how hard can it be to simply pick out a monitor for my computer?" The answer is the same as in most choices: it is simple if you know what you are doing and you know what to look for in a monitor. Otherwise, choosing the wrong monitor for the wrong reasons can be an expensive lesson.

What Kind of Displays Are Available?

In the world of IBM and compatible computers and displays, there are several types of monitors available. Most display monitor manufacturers emulate industry standard IBM models. In the PC market, these five types of monitors probably account for 99% all sales:

- Monochrome Display (MD)
- Color Display (CD)
- Enhanced Color Display (ECD)
- VGA Analog Display (AD)
- Variable-Scan Analog Display

The **IBM Monochrome Display** is the original display used with early PCs. It is capable of displaying crisp monochrome text or pixel graphics. It is usually driven by an IBM Monochrome Display Adapter (MDA, text only) or a Hercules Graphics Card (HGC, text or graphics), but an IBM Extended Graphics Adapter (EGA, text or graphics) can also be used.

The **IBM Color Display** was introduced at the same time as the IBM Color Graphics Adapter (CGA) and was the first color display offered by IBM for their PC line. It is capable of displaying 16-color text or graphics and is driven only by a CGA or EGA subsystem.

The **IBM Enhanced Color Display** was introduced with the IBM Extended Graphics Adapter and was part of IBM's attempt to standardize PC video. It allowed users to replace their MD+MDA, CD+CGA, or MD+Hercules video subsystems with the ECD+EGA combination. However, this display doesn't necessarily require an EGA video card; it can also be driven by any of the three previously accepted standard video cards: MDA, CGA, or HGC.

IBM's **VGA Analog Display** came onto the scene to support the Video Graphics Array (VGA). This display is quite a bit different than any of the three previous IBM displays in that it is the first one to accept color information in the form of pure analog signals. All others had color information brought in on digital signal lines, i.e., each color had a dedicated line into the monitor and was used to indicate presence or absence of the particular color.

With the advent of video resolutions greater than that offered by a VGA card, "SuperVGA", IBM 8514/A, and Texas Instruments 34010-based high-resolution video cards offering non-standard modes of operation became available and required monitors that supported these modes. NEC of Japan was one of the first monitor manufacturers to introduce a **variable scan analog monitor**, called the Multi-Sync. Its claim to fame was that it had the electronics necessary to automatically adjust itself and synchronize as necessary in order to display the non-standard video.

What's Your Application?

Before making any attempts at choosing a monitor, evaluate your situation and

determine exactly how you will use your system. Will you work primarily in a text-based environment such as writing line after line of code as a programmer, or will your needs be entirely different by spending most of your time in a graphical environment such as Microsoft Windows, Engineering CAD, or Desktop Publishing? Determine which video subsystem is best for your application according to evaluation factors such as performance and budget. Choosing a video subsystem will then mean choosing from the industry standard video adapters:

- Monochrome Display Adapter (MDA)
- Color Graphics Adapter (CGA)
- Hercules Graphics Card (HGC)
- Extended Graphics Adapter (EGA)
- Video Graphics Array (VGA) 8514/A

plus a few non-standard video adapters as well:

- SuperVGA (VGA w/extended resolution and color modes)
- Texas Instruments 34010 and 34020-based video cards

What Type of Monitor Do I Choose?

After choosing a video subsystem, it is necessary to choose the correct monitor type so that the video signals being sent out by the video card will show up on the display as expected! As mentioned earlier, certain video card and monitor combinations were made for each other. Picking a monitor is then simply a matter of ensuring that the one you get is the one specified for the adapter type chosen. However, things get a little more complicated when you start getting into the realm of non-standard video cards.

To ensure that the video card will play with a particular monitor, you must be certain that the electrical timing specifications are matched. The two most critical timing parameters are the vertical refresh rate and horizontal refresh rate. The hori-

zontal refresh rate is also called the line rate because it is the rate at which the monitor is capable of displaying "lines" of pixels, dots. More precisely, it is the rate at which the monitor scanning process moves from the right edge of the screen, retraces, and starts again near the left edge of the screen. Generally, most higher-resolution modes beyond standard VGA have unique line rates. The most common (and standard) line rate is that for a VGA analog monitor: 31.47KHz. This means that all VGA cards will output a horizontal sync signal that pulses 31,470 times per second, telling the monitor that the current line is finished and a retrace operation (to the next line) is necessary.

The vertical refresh rate is also called the frame rate because it is the rate at which the entire frame, or screen, is refreshed, i.e., the scan process is finished at the lower right corner of the screen and a retrace is initiated to start again at the upper left corner. Typically, frame rates are on order of 50-70 Hertz (times per second) although the industry is moving to higher rates to reduce the amount of "flicker" which causes eyestrain and fatigue.

Table 1 lists the monitor types that are used with industry standard video subsystems. When you get involved with non-standard video cards, you must know what the horizontal and vertical rates are produced by the card. You then need to purchase a monitor that supports the timing rates as produced by your video card! Failing to heed this simple rule will yield great disappointment and hassles! Keep in mind here that non-standard video cards often have peculiar timings associated with them and while virtually all of them require an analog monitor, not all analog monitors claiming compatibility with SuperVGA will support all SuperVGA modes of operation. It is *not sufficient* knowing that your video

card has a 800x600 resolution capability and the desired monitor also supports 800x600. The timing used by one monitor vendor for 800x600 may be different than the timing used by another vendor. Be careful! The safest way to avoid hassles like these is to choose a monitor that has multi-scanning capabilities, such as the NEC Multi-Sync series, instead of a monitor that has several discrete frequencies it is capable of seeing. These multi-scanning monitors are able to scan a frequency *range* and can lock onto any horizontal line rate within the prescribed range. Less expensive hi-resolution monitors are able to lock onto only specific, factory preset frequencies. The trick to using these less expensive monitors is to ensure that your video card utilizes a horizontal line rate that is exactly the same as one supported by the monitor. If it isn't the same, it probably will not work correctly!

What Should I Look For in a Monitor?

Once a video subsystem has been identified, and all of its associated timing parameters are understood, the time comes to select a monitor. So, what makes one monitor better than another? Whose monitor will be best for me and my application?

For most people, it seems that the primary criteria in selecting a monitor is price and performance. This isn't exactly a novel idea, but it certainly is accurate. Some may say that there are other reasons to choose a particular monitor, such as the availability of service, or product brand loyalty, but the bottom line is that for most of us, the single user, determining the best bargain for your buck is the key issue.

The cost of any particular type or model is easy to determine. Simply pick up any computer magazine, and see what the national mail order houses are asking. You will get the best price that way. Performance, however, is much more complex to evaluate.

Some of the items that may play an important role in your final selection are:

Screen

Size: How large of a screen area is needed? If your application tends toward text-based operations, then a smaller 12" or 14" screen is preferable over a larger one. Since text-based DOS applica-

tions will have the same number of columns and rows (i.e., 25 lines, with each line 80 characters wide) regardless of the size of the screen, using a large screen monitor effectively "spreads out" all of the characters over a larger area, resulting in characters that are not as crisp as when seen on a smaller screen. Most business graphics users, those that use Microsoft Windows, Lotus 1-2-3, and others, can usually get by with a 14" diagonal color VGA monitor. Serious CAD and Desktop Publishing (DTP) users are in a different category however, requiring much larger 19" or 20" diagonal screens. Often, these systems require specialized (and certainly non-standard) video subsystems and equally unique display monitors.

Screen sizes are measured diagonally and refer to the tube size, not the size of the viewable area. Typically, a 14-inch monitor will have a viewable area only about 12-1/2 inches diagonal.

Ergonomics: One of the more important qualities of a monitor is its ability to allow you to look at it for long periods of time without causing eyestrain and fatigue. Eyestrain can be caused by display flicker, poor contrast ratio, and glare. Some monitors will help combat glare by applying special anti-glare coating to the tube's face.

Flicker is more a function of the video subsystem driving the monitor, usually because the vertical timing rate is too small. One well-known example is IBM's 8514 subsystem. It drives the monitor in an "interlaced" fashion, meaning the image on the screen is painted there by the video subsystem writing all even numbered raster lines to the screen during one screen refresh, and then writing odd numbered raster lines on the subsequent screen refresh. Interlaced displays are used by some video cards for their higher 800x600 and 1024x768 resolutions. This is usually done to keep costs down. I recommend avoiding video subsystems that use interlacing techniques. Spend a few extra dollars and avoid this scenario altogether. It will be well worth the investment.

Controls: Practically every monitor has controls for adjusting the contrast and brightness. I have never seen a monitor without them, but if they are not available, don't waste your money. You need this feature!

There are usually other controls allowing the user to adjust the horizontal and vertical positioning of the image. These are important and it is wise to ensure that the desired monitor has them.

Some monitors also have a degaussing switch. This switch is really only necessary if the monitor will be left on indefinitely or for extended periods of time. What happens is that after long periods of time, the colors on color monitors may change somewhat, or there may be a slight distor-

Continued on Page 13

Video Sub-system	Horizontal Frequency	Monitor Type
MDA	18.43 KHz	MD, ECD
HGC	18.43 KHz	MD, ECD
CGA	15.75 KHz	CD, ECD
EGA	18.43 KHz	MD
	15.75 KHz	CD
	21.85 KHz	ECD
VGA	31.47 KHz	Analog(1)
8514/A	31.47 KHz	Analog(2)
	35.5 KHz	Analog(3)

Monitor Legend:

MD	IBM Monochrome Display or compatible
CD	IBM Color Display or compatible
ECD	IBM Extended Color Display or compatible
Analog(1)	IBM VGA Analog Display or compatible
Analog(2)	IBM 8514 Display or compatible: low-resolution mode: 640x480 non-interlaced
Analog(3)	IBM 8514 Display or compatible: high-resolution mode: 1024x768 interlaced

Table 1. Standard Video

Small Computer System Interface

*Nathan Baker
Zenith Data Systems
Technical Writer*

In 1986, Hewlett-Packard introduced an intelligent, general purpose 8-bit bus interface to be used with high performance mass storage devices. Originally designed for desktop and personal computers, this "small computer system interface" is referred to by its acronym, SCSI (scuzzy). The first SCSI interface (SCSI-1) approved by an ANSI committee became a standard (ANSI committee standard X3.131). This article primarily explains SCSI-1.

Since the implementation of SCSI-1, another ANSI committee has studied and approved a second SCSI interface standard. SCSI-2 is faster and more complicated than SCSI-1, and is intended for 16- and 32-bit bus architectures. Some information about SCSI-2 is provided here, in order to round out the material.

(Note: Hereafter, the term SCSI refers to SCSI-1 unless otherwise indicated.)

What is SCSI?

SCSI is an intelligent, bi-directional I/O bus interface. It does not define a device. SCSI is a set of interface standards that allow multiple compatible peripherals to share the same bus.

The SCSI standard put forth by the ANSI committee specifies the following:

- The physical and electrical characteristics of the SCSI bus
- The method for arbitrating bus contention between competing SCSI peripherals
- The timing parameters for SCSI bus operations
- The SCSI command set.

An explanation of the SCSI command set is not included in this article.

SCSI Peripherals and the SCSI Hardware Interface

A basic SCSI system is composed of a SCSI peripheral connected through a SCSI

cable to the host computer, which contains a SCSI processor. Since SCSI peripherals can have internal SCSI processors (although this is not a requirement of the SCSI specification), the SCSI processor in the host computer is often referred to as the host adapter.

The path for power and data transfer between the host computer and SCSI peripherals is through the SCSI cable. This shielded 50-conductor cable carries nine data lines, nine control/status lines, and 32 power and ground lines. The cable can be terminated in one of two configurations; single-ended or differential.

A single-ended SCSI cable uses the voltage level of a single wire (compared to ground) to indicate a logic level. A single-ended SCSI cable should only connect devices within a single cabinet, and can be up to six meters long.

A differential SCSI cable uses the potential difference between two wires to indicate a logic level. A differential SCSI cable can connect devices in different cabinets, and can be up to 25 meters long.

Although both single-ended and differential configurations use a 50-conductor cable, the pinout for the cable connector differs. Table 1 lists the pinout for both configurations.

The cable configuration dictates the type of peripherals that can be connected to the SCSI bus. Single-ended SCSI peripherals can only be used with a single-ended cable; differential SCSI peripherals can only be used with a differential cable.

The SCSI bus provided by the host computer can support up to seven SCSI peripherals. This bus is sometimes called the primary bus. If a SCSI peripheral has its own SCSI processor, it may provide a secondary SCSI bus. There could be up to seven secondary SCSI busses attached to the primary bus. Each secondary bus can

connect with up to seven SCSI peripherals, so the maximum number of SCSI peripherals controlled by a single host computer is 56. All the SCSI peripherals are controlled directly (if they are attached to the primary bus) or indirectly (if they are attached to a secondary bus) by the host computer.

It is possible to have multiple host computers connected to the primary bus. If this type of SCSI system is used, then optional extended SCSI commands can be incorporated and up to 2048 peripherals can be controlled by a single target peripheral under control of the initiator, which is one of the host computers.

The real beauty of SCSI is that even if you have 8, 56, or in the extreme case 2048 SCSI peripherals, the host computer only has to have one expansion bus slot occupied by the host adapter. The remaining expansion bus slots remain free, and can be occupied by other expansion cards (memory, ESDI, video, etc.).

SCSI Bus Operation

The host adapter provides the gateway between the host computer's memory and peripherals on the SCSI bus. Only two SCSI devices can use the bus at the same time. For example, a peripheral could communicate with another peripheral, or the host computer could communicate with a peripheral.

The SCSI bus can support parity, which is an option in the SCSI specification. If the system supports parity, the parity is odd, and when the parity does not match, an error is created or an operation is aborted. Some operations do not require parity, but if parity is being used, care must be taken so that the parity bit is not driven false. In most operations that do not require parity, the parity bit can optionally be set to a high-impedance state.

Each peripheral on the SCSI bus acts as

Table 1
SCSI-1 Cable Pinout (Single-Ended and Differential Configurations)

Pin Number	Single-Ended ¹	Differential
1	Ground	Shield ground
2	-DB0	Ground
3	Ground	+DB0
4	-DB1	-DB0
5	Ground	+DB1
6	-DB2	-DB1
7	Ground	+DB2
8	-DB3	-DB2
9	Ground	+DB3
10	-DB4	-DB3
11	Ground	+DB4
12	-DB5	-DB4
13	Ground	+DB5
14	-DB6	-DB5
15	Ground	+DB6
16	-DB7	-DB6
17	Ground	+DB7
18	-DB(P)	-DB7
19	Ground	+DB(P)
20	Ground	-DB(P)
21	Ground	DIFFSENS
22	Ground	Ground
23	Ground	Ground
24	Ground	Ground
25	N/C ²	TERMPWR
26	TERMPWR	TERMPWR
27	Ground	Ground
28	Ground	Ground
29	Ground	+ATN
30	Ground	-ATN
31	Ground	Ground
32	-ATN	Ground
33	Ground	+BSY
34	Ground	-BSY
35	Ground	+ACK
36	-BSY	-ACK
37	Ground	+RST
38	-ACK	-RST
39	Ground	+MSG
40	-RST	-MSG
41	Ground	+SEL
42	-MSG	-SEL
43	Ground	+C/D
44	-SEL	-C/D
45	Ground	+REQ
46	-C/D	-REQ
47	Ground	+I/O
48	-REQ	-I/O
49	Ground	Ground
50	-I/O	Ground

Note:

1. A minus symbol (-) in this column indicates an active-low signal.
2. Pin 25 is left open to protect the host computer's power supply. If a single-ended peripheral is accidentally plugged into a differential SCSI bus, it will not short TERMPWR (terminal power) to ground.

an initiator, a target, or both. Initiators send commands to targets, which then perform the command. To help system throughput, SCSI is designed to allow targets to disconnect from the bus, perform an operation, and then reconnect to the bus. While the

bus is free, an initiator can access another target or a previously accessed target can reconnect to the bus. This stand-alone operation requires a certain amount of intelligence on the target device, and not all peripherals can do it. If the device cannot

operate in stand-alone mode, it must remain attached to the bus for the remainder of the I/O transaction.

When a peripheral (target or initiator) requires attention, it connects to the bus. To determine which SCSI peripheral gets access to the bus in systems that have multiple SCSI peripherals, the bus uses arbitration.

SCSI Arbitration and Bus Access

Arbitration, which is an option in the SCSI specification, is a method of assigning unique ID numbers to SCSI peripherals that share the same bus, so that a distinction can be made between them. This gives peripherals the ability to poll the host adapter for attention in a prioritized manner. Arbitration also allows for multiple initiators on the same bus.

Simple SCSI processors usually do not support the arbitration option. In non-arbitrating systems, the peripheral cannot disconnect from the bus until the entire I/O transaction is finished. Non-arbitrating systems can have only one initiator.

Most sophisticated SCSI processors support arbitration. If the system has multiple initiators, or allows targets to disconnect and reconnect to the bus, then the system must support arbitration.

Even if the host adapter supports arbitration, not all peripherals can disconnect and reconnect; they must have enough on-board intelligence to allow them to decode and carry out commands on their own and to perform the disconnect/reconnect handshaking.

Bus access and arbitration in a SCSI system work in the following manner, which is simplified for the sake of clarity:

(Note: Throughout the rest of this document, signal names preceded by a minus sign (-) are active low.)

During system initialization, the host computer assigns a SCSI ID bit, which represents a number from 0 - 7, to each peripheral on the SCSI bus. The ID number is sent across DB0 - DB7. If DB0 is set, the ID number for that peripheral is zero, if DB1 is set, the ID number is one, etc. The ID numbers are prioritized, with seven being the highest priority and zero being the lowest.

To connect to the bus, the peripheral must wait until the bus is free, and then assert -BSY and drive its ID bit onto DB0 - DB7. If the peripheral sees a higher priority ID bit on the bus, it has failed arbitration and drops out of the running. Peripherals that lose arbitration de-assert their ID bits and release -BSY.

If the peripheral wins arbitration, it keeps -BSY asserted and asserts -SEL, and then becomes an initiator by driving I/O low. As an initiator, it needs to connect to a specific target. To alert and locate the target, the initiator drives the OR'd combination of its ID bit and the ID bit of the

target onto DB0-DB7, and de-asserts -BSY. The initiator waits a short time before looking for a response from the target.

The target determines if it is selected by monitoring -SEL, -BSY, I/O, and DB0-DB7. It is selected when its ID bit and -SEL are asserted, -BSY is de-asserted, and I/O is low. After it determines that it is selected, the target waits a short time, and then asserts -BSY. The target knows what initiator called it by examining the other ID bit that is on the bus. If more than two SCSI ID bits are on the bus or bad parity is detected, no targets will respond.

When the initiator detects an active -BSY signal, it releases -SEL, and the data bus can be used to transfer commands or data.

Peripherals that lose arbitration wait until the bus is free again, and then repeat the arbitration process. This goes on until the peripheral wins arbitration. This could mean that a device with a low priority might be ignored for a long period of time because a device with a higher priority needs frequent attention. It is the host computer's responsibility to see that this situation does not occur, by using rotating priority or a similar procedure.

SCSI Bus Phases

SCSI bus operations, including arbitration, connection, and reconnection, occur in the following phases:

Bus Free — Indicates no SCSI device is accessing the bus. This phase always follows a reset.

Arbitration — Allows SCSI devices to compete for access to the bus. This is an optional phase.

Selection — Allows an initiator to connect to a target.

Reselection — Allows a target that has disconnected from the bus to reconnect to the bus.

Command — Allows the target to request a command from the initiator.

Data I/O — Allows data transfer between target and initiator.

Status — Allows the target to send status information to the initiator.

Message (out/in) — Allows messages to pass between the target and the initiator. A message out phase sends information from the initiator to the target. A message in phase sends information from the target to the initiator. These two phases are optional.

Each bus phase is distinct and separate. The eight phases interact with each other, depending on the SCSI configuration, and if arbitration and messages are supported.

A typical SCSI operation that supports arbitration and messages uses bus phases in the following sequence:

1. Bus free.
2. Arbitration.
3. Selection.
4. Message out.

5. Command.

6. Message in.

7. Bus free.

If the peripheral is allowed to disconnect and reconnect, the following additional bus phase sequence can occur:

8. Arbitration.

9. Reselection.

10. Message in.

11. Data I/O.

12. Status

13. Message in.

14. Bus free.

The way SCSI bus phases interact, and the control signals that identify each phase, is explained in the following paragraphs, which expand on the bus phase sequence listed above:

1. All bus transactions begin with the bus free phase. This phase is detected by sensing the -BSY and -SEL signals. If neither signal is asserted for 400 ns (the bus settle delay time) or longer, the bus is free.

2. Devices wait 800 ns (bus free delay time) after they detect a bus free phase, and then begin the arbitration phase by asserting their ID bit and -BSY. If the SCSI system does not support arbitration, this phase is skipped and the selection phase occurs.

After 2.2 us (arbitration delay time), competing devices check the bus to see if there is a higher priority ID number asserted. If a higher priority number is present, the lower priority devices drop off the bus.

The device that wins arbitration asserts -SEL and waits for at least 1200 ns (bus clear delay time + bus settle delay time) for the losing devices to drop out and release their hold on -BSY.

3. The selection phase begins 400 ns (bus settle delay time) after -SEL goes active. The device becomes an initiator by driving I/O low, and then waits 90 ns (two deskew delay periods) before it de-asserts -BSY. Then, it drives the ID bit of the target it wants to communicate with onto the bus.

When the target determines it is selected, it asserts -BSY. The initiator waits 90 ns (two deskew delay periods), and if -BSY is still active, it releases -SEL and the data bus.

If the SCSI system supports messages, the initiator can send a message to the target at this point by asserting -ATN. If it does not support messages, the command phase is next.

4. If the target senses an active -ATN signal, it can enter the message out phase.

The target enters the message out phase at its own convenience, by asserting -MSG, driving C/-D high, and driving I/O low. The target asserts -REQ to request a byte of the message. The byte is sent by the initiator, which also asserts -ACK to indicate that the message is on

the bus. The -REQ/-ACK handshaking and byte transfers continue until the message is complete.

The message can tell the target whether or not the initiator can handle a target disconnection, and if it can handle a synchronous data transfer. The message can also specify a transfer period (see SCSI Timing Parameters). If the message is composed of multiple bytes, the initiator keeps -ATN asserted for the duration of the message.

5. The command phase begins when -ATN is de-asserted, and C/-D is high, I/O is low, and -MSG is asserted. The command (read, write, seek, etc.) is sent by the initiator to the target.

If the target cannot disconnect from the bus, or the SCSI system does not support arbitration or messages, the data I/O phase or status phase is next (see step 11 and step 12), depending on the operation.

6. A message in phase occurs if the target needs to send a message to the initiator. If the SCSI system does not support messages this phase is skipped and the bus free phase is next.

The target controls the message in phase by asserting -MSG, and driving C/-D and I/O high. The target asserts -REQ to indicate a byte of the message is on the bus. The byte is sent to the initiator, which asserts -ACK to indicate that the byte has been received. The -REQ/-ACK handshaking and byte transfers continue until the message is complete. When the message in phase is over, the target can disconnect from the bus.

7. The bus free phase occurs, and the host adapter waits for a peripheral to call for attention.

8. Another arbitration phase occurs when a new target or a previously disconnected target wants access to the bus.

9. If a previously disconnected target wins arbitration, the reselection phase can begin. The reselection phase is similar to the selection phase, except I/O is driven high. The target reconnects with its original initiator.

If the target that wins arbitration is not a previously disconnected target, the selection phase begins instead (step 3).

10. A message in phase can occur at this point, so that the previously disconnected target can verify the link with the original initiator. See step 6.

11. The data I/O phase occurs. The target and initiator use -MSG, C/-D, and I/O, to indicate the direction the data is transferred, and what the data is (a command, a message, or valid data). -REQ and -ACK serve as handshaking signals to control the transfer. See the following section for more information about data transfers.

12. The status phase usually follows at the

Table 2
Phase Coding and Transfer Direction

C/-D	I/-O	-MSG	Phase	Direction
0 0	0	Data out	Initiator	to target
0 1	0	Data in	Target	to initiator
1 0	0	Command		Initiator to target
1 1	0	Status	Target	to initiator
1 0	1	Message out		Initiator to target
1 1	1	Message in		Target to initiator

end of the data I/O phase. This phase is similar to a message in phase. The target drives C/-D and I/-O high, and de-asserts -MSG. Then the target sends a status report to the initiator indicating the status of the last operation. The report can be a single byte or multiple bytes. -REQ and -ACK handshaking control the transfer of the report bytes.

13. The target enters a message in phase by asserting -MSG. A message in phase always follows a status phase. This message tells the initiator the command is complete and the target is going to disconnect from the bus. The target is finished with the operation. If the SCSI implementation does not support messages, this phase still

occurs, but the only valid message is the command complete message.

14. The bus enters the bus free phase again, and the host adapter waits for a peripheral to call for attention.

The phase sequencing can be altered by two asynchronous events, an active -ATN or -RST signal. When an initiator asserts -ATN an attention condition is created, and the target knows that the initiator has a message waiting for it. The target enters the message out phase at its convenience, and then gets the message and acts on it accordingly.

If -RST is asserted and held for a minimum of 25 microseconds (the reset hold time), a reset occurs. There are two ways a SCSI implementation can handle a reset: hard or soft. For a hard reset implementation, the following reset sequence occurs:

1. All uncompleted commands are cleared.

Table 3
SCSI Bus Timing Parameters

Parameter	Period	Definition
Arbitration delay	2.2 microseconds	The minimum time a peripheral waits, after it has asserted -BSY, before it examines the bus to see if it has won arbitration.
Assertion Period	90 ns	During synchronous data transfers, this is the minimum time a target keeps -REQ asserted, and the minimum time an initiator keeps -ACK asserted.
Bus Clear Delay	800 ns	The maximum time for a peripheral to stop driving all bus signals after a reset, after -SEL is asserted by another peripheral during arbitration, and after the bus free phase is detected. ¹
Bus Free delay	800 ns	The minimum time a peripheral waits after it detects a bus free phase, before it asserts -BSY and enters the arbitration phase.
Bus Set Delay	1.8 microseconds	The maximum time a peripheral has to assert -BSY and drive its SCSI ID onto the bus, after it detects a bus free phase.
Bus Settle Delay	400 ns	The minimum time to wait for the bus to settle before any control signals change states.
Cable Skew Delay	10 ns	The maximum difference between the propagation time of any two bus signals when measured between any two peripherals.
Data Release Delay	400 ns	The maximum time for an initiator to release the data bus following a transition of I/-O from low to high.
Deskew delay	45 ns	The minimum time for deskew of signals.
Hold time	45 ns	The minimum time data can be on the bus and read by an initiator or target after receiving handshaking pulses during synchronous data transfers.
Negation Period	90 ns	The minimum time a target de-asserts -REQ, and an initiator de-asserts -ACK, during synchronous data transfers.
Reset Hold Time	25 microseconds	The minimum time for -RST to be asserted.
Selectin Abort Time	200 microseconds	The maximum time a peripheral takes after it has been selected, before it asserts -BSY.
Selection Timeout Delay	250 milliseconds	The minimum time a peripheral should wait for an active -BSY signal after the selection phase before performing a time-out procedure. ²
Transfer Period	>= 200 ns ³	The minimum time between leading edges of successive -REQ pulses and successive -ACK pulses during synchronous data transfers.

Notes:

1. The maximum time for a peripheral to clear the bus after detecting a bus free phase can be stretched to 1200 ns, when the bus settle delay time is added.
2. 250 milliseconds is the recommended period. The selection time-out delay is determined by the specifications for each peripheral.
3. Greater than or equal to 200 ns. The transfer period can be set during the message phase. If messages are not supported, the system relies on the other timing parameters to determine the minimum time between -REQ and -ACK pulses during synchronous data transfers.

Table 4
Fast SCSI Timing Compared with SCSI-1 Timing

Parameter	SCSI-1	Fast SCSI
Assertion period	90 ns	30 ns
Negation period	90 ns	30 ns
Cable skew delay	10 ns	5 ns
Deskew delay	45 ns	20 ns
Hold time	45 ns	10 ns
Transfer period	200 ns	101-199 ns or more

- All SCSI device reservations are cancelled.
- SCSI operating modes return to their default conditions.

A soft reset implementation allows devices to reset themselves and not affect other devices on the bus. For a soft reset implementation, the following reset sequence occurs:

- Uncompleted commands are attempted and completed if they were fully identified.
- All SCSI device reservations are preserved.
- Current operating modes are preserved.

In either reset implementation, a bus free phase follows the reset.

SCSI Transfer Operations

Transferring data, commands, and messages between initiators and targets is controlled by five signals. The -MSG, C/-D, and I/-O signals control the direction and type of transfer, and -REQ and -ACK provide handshaking. The phase coding of the three control signals, and the direction the data flows, is shown in Table 2.

The SCSI bus can transfer commands, messages, and data, in synchronous and asynchronous modes.

A synchronous transfer can only occur

the target prior to -ACK being asserted by the initiator. The target does not wait for an -ACK from the initiator before it asserts -REQ, which indicates that it is ready to send or receive the next byte of data. The target continues asserting -REQ (and sends or receives a byte of data) until it reaches the maximum -REQ/-ACK offset. Then the target compares the number of requests (-REQs) it made with the number of acknowledgements (-ACKs) it received to make sure it transferred all the data. Synchronous data transfers can occur at a rate of up to 4 megabytes per second.

Asynchronous transfers are much slower than synchronous transfers, but can be used during all phases. In asynchronous mode, the target uses the I/-O signal to control the direction of the transfer. When I/-O is high, the direction is from the target to the initiator. When I/-O is low, the direction is reversed. During asynchronous transfers, a request signal is followed by the transfer of a single byte, which is followed by an acknowledge signal. This REQ/byte/ACK handshaking occurs for every byte (command, message, or data) until the transaction is complete. The maximum rate for this type of transfer is 1.5 megabytes per second.

Regardless of the transfer mode, data

during data I/O phases, and only if the target and initiator agree upon it during the first message phase. They must also agree on a minimum period between the -REQ and -ACK handshaking signals, and the maximum -REQ/-ACK offset. The -REQ/-ACK offset determines how many times -REQ can be asserted by

on DB0 - DB7 must be present for at least 45 ns (hold time).

Note that some SCSI manufacturers have pushed SCSI data transfer rates to the limit, by shortening cables and using faster parts, and have achieved synchronous transfer rates up to 5 megabytes per second and asynchronous transfer rates up to 2.5 megabytes per second.

SCSI Bus Timing Parameters and Fast SCSI

Table 3 lists and defines the timing parameters for the SCSI bus.

Fast SCSI is part of the SCSI-2 specifications, but can be used as an option in SCSI-1 implementations that use differential cables, if the hardware can handle the decreased timing parameters.

Table 4 compares the timing differences between Fast SCSI and SCSI-1. The other SCSI-1 timing parameters can remain the same.

If the hardware can handle the increased speed, the Fast SCSI option can be set during the message out phase by setting the transfer period to any value less than 200 ns and greater than 100 ns.

Fast SCSI systems can transfer data at a synchronous rate of up to 10 megabytes per second. Asynchronous data transfer is not allowed.

SCSI-2 and Son of SCSI

Since many optional commands are part of the SCSI-1 specifications, there has been a need to further standardize the ANSI specifications. SCSI-2 addresses this need.

Even though SCSI-2 is fresh and new, SCSI-3 is already being planned. In future articles I'll examine SCSI-2, and SCSI-3, eventually. For now, have fun with SCSI-1.

✽

Continued from Page 8

tion in the screen image. This phenomenon is due to metal parts within the monitor (specifically the monitor's "shadow mask") becoming magnetized. The resultant magnetic fields can adversely affect the electron beams within the CRT and you see the result as a lousy image. The monitor's degaussing circuitry removes this magnetic buildup. A switch is not necessary for most applications because virtually all monitors will perform this degaussing operation every time they are powered on.

Whose To Buy? Where To Buy From?

There are a myriad of vendors from which to choose your monitor. Computer equipment dealers and mail order houses entice you with great deals on a Sony, NEC, Samsung, Amdek, and other little known or generic brand monitors. As the old adage warns: caveat emptor, or buyer beware! You usually won't go wrong choosing one

of the big boys' models, but you might get burned by the no-name brand, or the super dirt-cheap models. I've been privileged to see some of the low-end, inexpensive monitors under evaluation by Zenith engineers, and the sight is not pretty. I know where all of the rejected junk goes: straight onto the mail-order heap! This doesn't mean that all of the equipment available through the mail is junk, but some of it certainly is. If you want to avoid getting burned, buy only what you know is good, and from an outfit you can trust.

The lion's share of computer monitors sold these days are of the VGA analog variety. VGA video cards can be had for less than \$100 so a VGA card and a VGA analog monitor are within the budget of most computer hobbyists.

As for whose monitor to buy, my choice, by a wide margin, is Zenith's ZCM-1492 FTM. It has an absolutely beautiful screen with deep, rich colors on a jet black

background, virtually no glare, and is vastly superior to anything else on the market. This monitor has consistently received rave reviews from most of the PC-oriented magazines and is a very popular choice. You could buy another model from another manufacturer, but they all seem to look the same. Very little distinguishes one from the other. The one that does stand out above all the others however, is the Zenith. It makes an excellent choice.

So choosing a monitor is really quite a simple task, the only time it isn't straightforward is when you are dealing with non-standard video cards and their timing peculiarities. Just remember that standard monitors will support the standard video cards, and be careful with the non-standard video. Make sure your monitor supports the *timings* for the non-standard modes you need. If you keep this mind, you then should have no trouble at all making your video card and video display work together. ✽

QUIKDATA - 15 YEARS OF H/Z SUPPORT!

YOUR H/Z ENHANCEMENT EXPERTS!

ACCELERATE YOUR PC/XT/AT!

From Sota Technologies, Inc., the fastest and most proven way to give new life to your H/Z PC/XT computer, giving it -AT compatible speeds! Turn your turtle into a -286 rabbit with a 12.5 Mhz 80286 or 80386 16Mhz SX accelerator board. Complete with 16K on-board CACHE.

The EXP12 **286i** is the effective solution, making your H/Z150/160/150/158/159 series of computers, or any general PC/XT computer faster, in many cases, than a standard IBM AT type computer! **You won't believe your stop watch!**

EXP-12 - \$275

EXP386 - \$449 Much faster 16Mhz 80386 SX version

For your H/Z241 or 248 we have the MicroWay 20Mhz 386SX accelerator which plugs into a slot and cables to the CPU. Run enhanced mode software such as Windows V3. 32K Cache with optional 32K add-on for 64K cache. Landmark 27, Norton SI 22.4. Cable extra, specify which is needed.

MWFCACHE - \$450 20Mhz 386SX Accelerator card

MWCLCC - \$95 LCC interconnect cable

MWCLCC - \$95 PLCC interconnect cable

MWCA32 - \$65 32K Cache add-on option

387SX20 - \$149 Optional 20Mhz coprocessor

MEMORY UPGRADES

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

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

EAZYRAM - \$89 Upgrades EaZy PC from 512 to 640K

ZMF100 - \$55 Will allow you to upgrade your H/Z110/120 (old motherboards; with p/n less than 181-4918) to 768K system RAM. Requires 27 256K DRAM chips.

Z100MP - \$55 Similar to ZMF100 above, but for new motherboards with p/n 181-4918 or greater.

3MB RAM BOARD for Z241/248 computers is an excellent memory card. Will backfill your 512 to 640K, and provide both extended and expanded RAM; all can coexist. Uses 100ns M256-10 RAM chips, 36 per megabyte desired. Minimum of 18 DRAM chips required (\$1.95 each).

EVATRD - \$119

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

Z605-1 - \$149

Z386/20, Z386/25, Z386/33, Z386 EISA 2MB SIMM 80ns UPGRADE to add increments of 2MB to these systems. Two required.

ZA3600ME - \$89

ZA3800MK - \$349 4 megabyte SIMM upgrade for Z386/20, /25, /33, /33E. Must have 4-1 meg SIMMs installed first.

WINCHESTER UPGRADE KITS

PCW20 - \$239 Complete MFM winchester setup for a H/Z150, 148, 158, 159, 160, PC etc. Includes 21 meg formatted half-height Segate ST-124 37ms drive, controller, cable set, doc.

ST-124 - \$195 Bare drive only

PCW30 - \$295 32 meg with 28ms Segate ST-138-1.

ST-138 - \$249 Bare drive only

PCW40 - \$319 42 meg with 28ms Segate ST-251-1.

ST-251-1 - \$269 Bare drive only

PCW80 - \$595 80 meg with Segate ST-4096 full size drive.

ST-4096 - \$549 Bare drive only

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

WDCON - \$59 PC/XT hard drive controller board

WDATCONF - \$95 1:1 interleave HD/floppy controller for AT's

FLOPPY DRIVE SAMPLE

MF501 - \$71 5" 360K DS/DD drive

MF504 - \$79 96 TPI 1.2 meg AT/Z100 drive

MF353 - \$79 720K 3.5" drive in 5" frame

MF355 - \$85 1.4 meg 3.5" AT drive in 5" frame

TM100-2R - \$69 40tk DS reburb (H8/89/Z100 PC type)

Also other drives and full line for older systems

ADD AN EXTERNAL HARD DRIVE TO ANY LAPTOP OR PC DESKTOP with our EXPORT hard drive. Plugs into a parallel port (do not lose the port) to give you an affordable way to easily add a hard drive to your computer. Fast!

EWIN20 - \$495 20Mb unit

EWIN40 - \$625 40Mb unit

ANY DRIVE IN YOUR PC/XT/AT

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

CCARD2 - \$79 CCARD - \$99

CCARD4 - \$125

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

BPACK2.8 - \$319

BPACK1.4 - \$269

BPACK1.2 - \$269

BPACK360 - \$269

8-BIT/Z100

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

OTHER STUFF

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

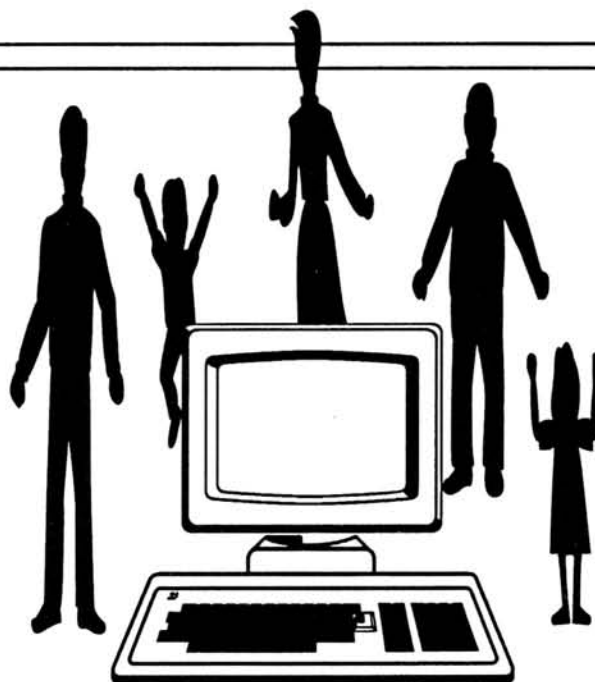
Call or write in to place your order, inquire about any products, or request our free no obligation catalog. VISA and Master Card accepted, pick up 2% S&H. We also ship UPS COD and accept purchase orders to rated firms (add 5% to all items for POs). All orders under \$100 add \$4 S&H. Phone hours: 9AM-4:30PM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

QUIKDATA, INC.

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

All Around the eaZy PC

Laszlo M. Uesei
420 Philip Road
Niles, MI 49120



Suppose you are running a business and find that one of your promising ventures will result in a substantial loss. What is the best strategy to follow? Why, you try to minimize the losses!

The eaZy PC was Zenith Data Systems' effort to put a computer into every household. Unfortunately, not enough households had a pressing need of a computer to make this venture profitable. For this reason, these machines became widely available through salvage houses and the Bargain Centre. Smart Huggies (Zuggies? Bullies?) bought them up at truly bargain prices. Now that so many of us have them the question is what can we do with them? The answer is: plenty. The eaZy PC is a very good general purpose machine in its own right. There are a few areas though, that can — and perhaps should be — improved.

Before continuing with these improvements, I have to give credit where credit is due: to the active members of the ZUG

COM1 Bulletin Board. Most of the ideas and improvements are the brainchildren of this august group(e).

Now let's see what we have: First, take a look outside of our eaZy PC before we try to do anything in way of improvement. There are 3 varieties of these machines:

1. one floppy,
2. two floppies, and
3. one floppy and one hard drive.

How to add a hard drive for those that have none, will follow later. For those with two floppies, it is possible to add a hard drive and even install connection for a third, outside floppy, perhaps a 5-1/4". More about this later, too.

Now let's look at the backside. There are two DB type connectors: One 25-pin and one 9-pin, the latter is marked 'MOUSE'. There is also a 64-pin, dual-in-line connector, covered with a small plastic cover. Let's see what these connectors are good for:

1. The RS-232/9-pin is a half-witted serial

port. The pinout:

Pin	Function	Pin	Function
1	NC	6	NC
2	RXD	7	RTS
3	TXD	8	NC
4	DTR	9	NC
5	Ground		(NC=not connected)

Regrettably, CD, DSR, CTS, and RI are missing; therefore, some devices cannot communicate with the eaZy PC.

2. The 64-pin expansion connector is intended for the addition of a "dongle". About dongles later. See Figure 1 for the 64-pin expansion connector pinout.

Row "a" is on top, row "b" on the bottom, pin #1 near the DB plugs.

This pinout is especially dedicated to those heroic souls who are able and willing to build their own expansion device. Such an heroic feat, however, is outside the scope of this article.

3. The RS-232/25-pin is a standard IBM-type parallel port. With the appropriate

Pin	Function	Pin	Function	Pin	Function	Pin	Function
a1	I/O CLK	a17	A14	b1	GND	b17	DACK0*
a2	D7	a18	A13	b2	RST DRV	b18	DRQ0*
a3	D6	a19	A12	b3	+5VDC	b19	REFRQ*
a4	D5	a20	A11	b4	IRQ2	b20	PCLK
a5	D4	a21	A10	b5	-5VDC	b21	IRQ7
a6	D3	a22	A9	b6	DRQ1	b22	IRQ6
a7	D2	a23	A8	b7	-12VDC	b23	IRQ5
a8	D1	a24	A7	b8	MSPEK	b24	IRQ4
a9	D0	a25	A6	b9	+12VDC	b25	IRQ3
a10	XTRReady	a26	A5	b10	GND	b26	DACK1*
a11	AEN	a27	A4	b11	MEMW*	b27	END/TC
a12	A19	a28	A3	b12	MEMR*	b28	ASTB
a13	A18	a29	A2	b13	IOW*	b29	+5VDC
a14	A17	a30	A1	b14	IOR*	b30	OSC
a15	A16	a31	A0	b15	DACK2*	b31	GND
a16	A15	a32	GND	b16	DRQ2	b32	GND

* marks overscored (that is reversed) symbols

Figure 1

cable, any parallel printer or other peripheral will work from this port. Since the pin designations are the same as on any other parallel port, I did not go into these here.

Now for those whose ideal is Richard the Chickenhearted: Do not turn the page in disgust — yet. There is hope for you!

At this point, a warning is appropriate: If you are squeamish about surgical operations on your computer, seek out a specialist (after all, who is willing — or able — to take out his own appendix?). The ZUG COM1 Bulletin Board is an excellent source to find one in your area.

The soldisant serial/mouse port (I just had to put in some French to honor Groupe Bull Cie.) is a cripple. Nevertheless, some software and peripherals, notably modems and communication programs, can use it "as is". Indeed, I made up a modem cable (Figure 2). Using this cable and an ADC modem, I was communicating with the ZUG bulletin board flawlessly at 1200 Baud. However, another modem did not work at all. The software was the same, Procomm 2.42 in each case. Reportedly, BOYAN will also work.

with a little bit of luck, one still can be found. The third kind, unfortunately, is practically non-existent. Of course, it is always possible to connect a modem to the second kind if phone communication is the avowed purpose of your natural life.

The dongle, whichever model is at hand, is a small sheet metal box with a flange. The flanged side has an opening out of which a short ribbon cable is dangling. At the end of the cable, there is a 64-pin, double-row connector to fit the pins inside the eaZy PC. Push this connector carefully, but firmly, onto the pins. Tuck the excess cable back into the dongle, there is plenty of room inside for this. The flange is 'supposed' to have two captive screws to fit two tapped holes on the back of the eaZy PC. Bolt the dongle to the eaZy PC, making sure that the ribbon cable is not pinched. Voilà!

Fate is fickle: If you buy a dongle, you get it "as is": meaning that there are some faulty ones around. These dongles are reputed to short out the memory chips inside them. Caveat emptor!

FBE Research advertises an extension device in REMark. It has the memory exten-

Centre was practically giving them away. If you did not get one, FBE Research's Smart-Watch will work just fine (address at the end of this article). This procedure requires minor surgery, so if you have second thoughts about opening the cabinet, this job is not for you. The more hardy souls will proceed as follows:

Before you perform any surgery, make sure that GOOD tools are at hand. You will need a GOOD #2 Philips screwdriver. The one in the local department store or the corner hardware store is NOT good enough. Get one from a shop specializing in tools for mass production. It is certainly worth the price.

Put an old blanket on your worktable. It will protect the eaZy PC from scratches, not to mention the teakwood tabletop's mirrorlike finish. Next, lay the eaZy PC gently on its right side and locate the 4 screws holding the cabinet bottom. These screws are located near the corners in the SMALL dimples (not in the 6 big ones!). These must be removed and the bottom, which contains the computer and the disk drive(s), can be now separated. You did not forget to have pencil and paper handy to note locations, wire colors and such, so that all that stuff can be put back together correctly afterwards, n'est-ce pas? Also, a small container is nearby to hold all small stuff until reassembly time, nicht wahr? (Too much French, gotta use some German tool!) There are wires with plugs connecting the bottom to the rest of the eaZy PC. These must be unplugged and the plastic 'mousetails' (cable ties) holding one cable cut. The speaker must be taken out completely before the sheet metal cover can be removed, baring the soul of the machine. The clock goes under the system ROM, so this animal must be located now. It is a 28-pin device in a socket near the dongle connection, bearing the part number 444-613-4. Pull it carefully out of the socket and plug the SmartWatch into its place, keeping in mind the orientation of the slot. Now, plug the system ROM into the SmartWatch, correctly orienting it. Inspect your work CAREFULLY for pins bent under or otherwise not entering the socket.

All that remains is to put everything back together the way it was before. The sketches and notes you made during disassembly come in handy now. Make sure that the plugs you unplugged before are pushed all the way in and correctly oriented. If you have some extra mousetails use them, but the eaZy PC will work without them. Fastening the bottom back to the rest of the machine requires the skill of a minor 'jongleur', but it can be done.

A possible complication: Removing the four screws holding the cabinet bottom produces sometimes unexpected results. The bottom is a sheet metal pan, while the mating part is molded of plastic. There are 4 spacers screwed into the 4 plastic bosses

eaZy PC
DB-9

Modem
DB-25

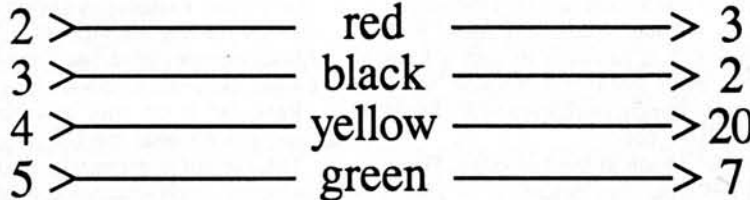


Figure 2. Modem Cable.

Many peripherals DO need a GOOD serial port. Therefore, the first improvement is to add one. Fortunately, it is so easy, that even the all-thumbs among us can do it.

First however, one must steal, buy, rob, or borrow (and never return) a "dongle". Originally, Zenith Data Systems provided 3 kinds of dongles:

1. ZDS model EZA-14 contains 128 K memory only, to bring the total memory up to 640 K.
2. ZDS model EZA-15 has the above mentioned memory, plus a GOOD serial port.
3. ZDS model EZA-11 contains all of the above, plus a 1200 Baud modem.

Alas, Zenith Data Systems quit making these dongles years ago. There are quite a few of the first kind around in the salvage houses, though (some addresses will follow at the end of this article). The second kind is scarce, like Edsel dealers. However,

and room for one more device: either a GOOD serial port or a real time clock (but not both). It is a quality product.

Life is sometimes full of surprises, some good, some otherwise. What if in the just described procedure one or both of the captive screws are missing? Do not despair! No situation is utterly without hope. You will need a 5/32" (.156") drill to enlarge the hole on the dongle's flange, (NOT in the eaZy PC!) where the captive screw is missing. Now get some #6-32 screws at the corner hardware store, about 1/2" long, and you are in business.

Minor Surgery

According to an old adage, the owner of a clock knows the time, the owner of more than one clock is never sure. The eaZy PC does not own a real-time clock (comment s'appell: une horloge, eh?). So next we will enrich our eaZy PC with an "horloge". Not so long ago the Bargain

on this molding. These spacers should remain in the plastic when the bottom-holding screws are removed, but sometimes they do come out. Also, *horribile dictu* (that's Latin!), the boss itself may be broken. Judicious use of that great American standby, the Magic Glue, (Cyanoacrylate to you, and for heavens sake, do not even think of using Elmer's glue or library paste) will save the situation. *Probatum est*. Just do not glue your fingers together or to the computer.

Hopefully, you did all the preceding correctly and now it is time to boot up your pet. It should boot up and show the time and date completely off the wall. (If it does not boot up, you need more help than what I can provide sight unseen.) The SmartWatch comes with software on a 5-1/4" disk, which does not fit the eaZy PC's 3-1/2" drive. (What else did you expect?) You will have to find a way to copy it onto a 3-1/2" disk before you can set the correct date and time.

Open Heart Surgery

The next improvement is the installation of a hard drive inside the cabinet. **Warning:** Unless you are sure of your surgical talents, your sheet metal butchery skills, and are of fearless disposition, don't even begin! This operation is definitely not for the faint of heart or the all-thumbs person. (Get that GOOD Philips screw-driver!)

Before you begin, use your eaZy PC as is to make yourself familiar with its idiosyncrasies, while you acquire the necessary parts. There is no point in having your pet open to the elements, nor to stay computerlorn for weeks while you are waiting for some stuff on backorder.

As the hard drive does not fit the existing floppy drive mounting holes, an adapter is necessary. Being averse to drilling holes into any electronic device (once I drilled into a live power transformer!), I do not plan to use the Zenith Data Systems HD mounting bracket, part N° 012-09717, because it necessitates to accurately locate and drill 4 holes in the bottom pan. Nor do I deem it necessary to install a cooling fan. Several people who installed hard drives did not experience any cooling problem without a fan.

The bracket can be fabricated out of a piece of sheet metal without disassembling the computer yet. This is where the sheet metal butchery skills come in handy. Take your time and make it accurate. See Figure 4.

The next ingredient is the ribbon cable. Some people are hesitant, although making up the cable is probably the easiest part of the whole operation, provided you have a small vise.

Needed ingredients are a 40-conductor ribbon cable, about 7" long, and two 40-pin IDC socket connectors. If the local

eaZy PC CPU Board

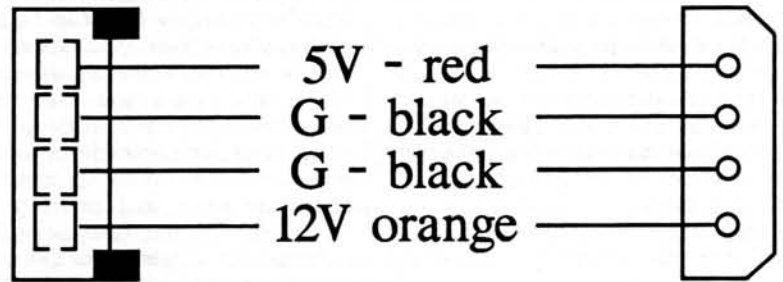


Figure 3. HD Power Cable.

Radio Shack or other electronics store does not carry them, they are available from James Electronics or Digi-Key or JDR Electronics (see end of article).

Procedure:

1. Insert the cable end into the opening of the socket connector. One end of the connector is marked with a small triangle. The colored edge of the cable is located toward this end.
2. Put the connector into the vise and compress it until the connecting hooks click into engagement. Takes a LOT of pressure.
3. Do the exact same thing with the other

end of the cable. It wasn't too difficult now, was it?

After the cable is made up, you have two choices, depending on the hard drive you've got: For some drives it has to be folded to the shape similar to the floppy cable between the main board and the floppy drive "A", for others leave it straight.

Western Digital model WD93028 needs the cable folded, Miniscribe model 8225XT needs it straight. I did not run across other hard drives.

The plastic cover plate, if one is furnished with the hard drive can be discarded. Likewise, the connector for the

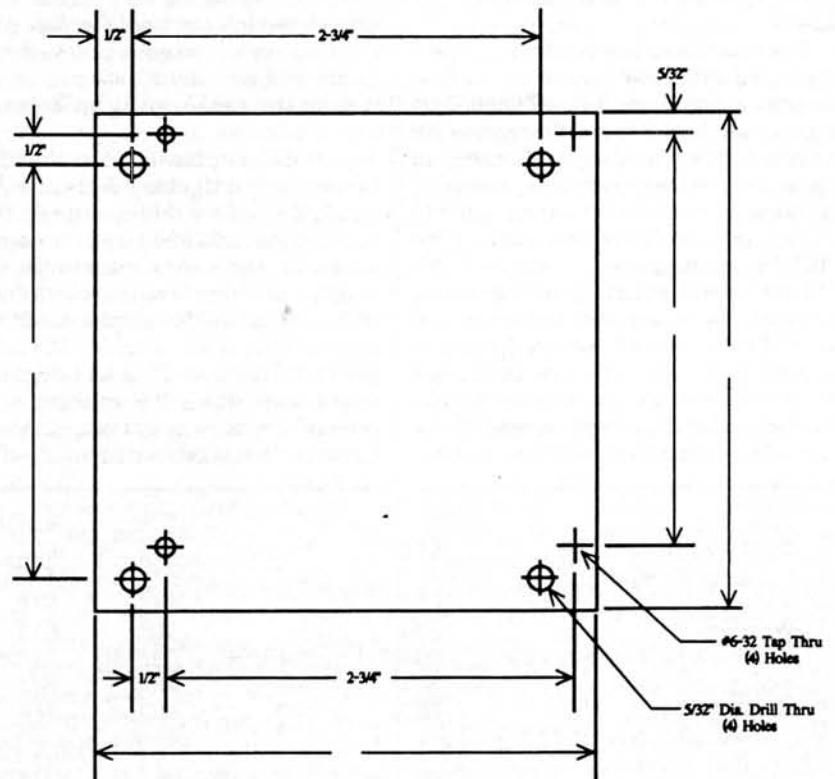


Figure 4. HD Adapter.

activity LED, or the cable thereof: Cut it off or tape it up so it cannot get into mischief. The small, original signal panel will signal the HD activity.

The next item is the power cable for the hard drive. Now, this one is a bit more of a bother. If you have a crimping tool or you are handy with the soldering iron, you can fabricate it yourself. However, you can try to get it all made up from Zenith Data Systems, if you do not mind the price. The ZDS cable has a connection for the cooling fan, which can be left dangling in the breeze or cut off.

If you decide to fabricate it yourself, you will need the following material (see Figure 3):

1. One 4-pin housing, AMP # 480426
2. One 4-pin housing, Waldom # 22-01-3047
3. Four contact sleeves, AMP #60619
4. Four contact springs, Molex # 08-50-0114
5. 4 pieces of #22 stranded cable, 4" long, one red, one orange and 2 black ones.

Remove about 1/8" of insulation from the ends of each wire. Crimp and solder a contact spring to one end and a contact sleeve to the other end of each wire. Now push the springs into the small housing and the sleeves into the larger housing until they click into place. Make sure that the right colors go to the right place. Should you make an error, don't curse. With the help of the Good Lord and a VERY SMALL screwdriver the contacts or the sleeves can be disengaged, pulled out of the housing and re-inserted into the correct hole. See Figure 3.

The most important (and most expensive) ingredient is the hard drive itself, as expected. To my knowledge Zenith Data Systems used any of four different brands: Western Digital, Seagate, Miniscribe, or Conner. Any of these will work, suit yourself. These are all IDE-XT drives, 20 MB capacity. (Miniscribe is now part of the MAXTOR organization).

I mention here those greedy souls, who would like to try a 40 MB drive. Yes, such IDE-XT drives are rumored to be in existence, (although I've never seen one), but you will need special software to take advantage of these. Furthermore, these drives will not boot, you will have to boot

from a floppy.

The system ROM is set up to accommodate any of 4 slightly different drives. See Figure 5.

Once you have the hard drive and all the other parts ready, the time is here for open heart surgery. Open up the computer as described earlier. Most of the eaZy PC's have a second shield over the disk drives. This is held by 4 screws from below. Remove this shield, it will not be reused. Then disconnect the cables from drive B and take out the 4 screws holding the drive. These are the ones located in the large "dimples". For the time being tape the ribbon cable so it is out of the way. If you do not intend to add an outside drive, you can even cut it off near the drive "A", using a pair of sturdy scissors. Pull off the drive power cable from the main circuit board and discard it. (If you are of Scottish ancestry you may want to rework this power cable to fit the hard drive, instead of fabricating a new one.)

Some eaZy PC-s have a 40-pin double row connector for the hard drive on the main circuit board, others have only the holes provided for it. If yours is neither of these, quit right there. It is an older model and adding a hard drive to these is not in the book of Fate. If yours has the holes only, you encountered a minor surgical emergency. The board must be removed to have access to the bottom side. Unscrew the 6 hex-shaped spacers and the small, hex-shaped extender screws at the two port connectors. A 1/4" and also a 3/16" nutdriver would be very handy here, although the job can be done with a pair of duckbill pliers in an emergency. The circuit board will be easy to lift out, once you unplug the cables going to floppy drive "A".

The electronic suppliers already mentioned carry a double row header, which you will need for this operation. The 3M part number is 929665-09-36. It comes in a long stick and must be cut to the correct length. Insert the shorter side into the series of holes and solder all pins CAREFULLY. Inspect your work, resolder all suspicious joints and remove all solder bridges which might cause shorts. If everything looks OK reinstall the main circuit board. A word of caution: the silkscreened marks for this

header on the circuit board are backwards. Mark 40 is actually #1 and pin marked 1 is actually #40.

Fasten the adapter plate to the bottom of the hard drive and connect the new cables to the drive. It is easier to do it at this time, before the drive is fastened to the bottom pan. Then using the small screws removed from the floppy drive, bolt the hard drive into place. Now connect the drive cables to the main board.

There are a couple jumpers to be set correctly:

1. Jumper W-6 tells the system the number of floppies. ON=1 floppy, OFF=2 floppies. This jumper is located next to the power cable plug for the "B" drive.
2. Jumper W-18 OFF means hard drive present, ON means no hard drive installed. It can be found between the V-40 (smaller square) chip and the next row of chips toward the rear.

Leave any other jumpers the way they are.

Before I'd go any further at this point I'd check the patient for vital signs. The cables dangling from the upper part of the machine are long enough to connect the two parts temporarily, while the top part is laying on its side and the bottom pan is standing on its legs, right side up. Plug in the keyboard and turn the machine on. You should not notice any smoke anywhere and the hard drive should spin up. If all is well sew up the patient unless you want to add an outside floppy drive.

For this addition you will need a drive in a housing, complete with its own power unit. It could be even a 5 1/4" drive. The easiest way is to chop an opening into the plastic top of the base where the ventilation slots are. Stick the ribbon cable, that used to go to drive "B", thru the opening thus created. You will then need a 34-wire floppy extension ribbon cable with a pin connector to match the old "B" floppy connector on one end and a card edge connector on the other.

If the vital signs are OK, sew up your patient and try to boot again.

The pruders among us will insist, that the gaping hole in the side of the bottom pan, where the "B" floppy used to be, constitutes indecent exposure. While leaving it open does not do any harm, it is easy to cut a piece of sheet metal to cover the eaZy PC's nakedness and attach it with Magic Glue.

Post-Operative Care

Assuming that every little thing is working now, boot the eaZy PC with a floppy from drive "A". In my experience any MS-DOS later than Ver. 3.21 will give unexpected and/or unpredictable results, but then my exposure to eaZy PCs is limited to only a few of them and my DOS 4.0 may be bugged. Rumor has it that PREPPing an IDE

Continued on Page 32

Type:	0	1	2	3
Total Cylinders:	612	612	612	612
Number of Heads:	4	4	4	4
Reduced Write Current Cyl.:	615	615	615	615
Precomp. Cyl.:	128	0	300	615
Control Byte/ECC Span Byte:	03/OBH	03/OBH	03/OBH	13/OBH
Removable:	NO	NO	NO	NO
Servo Wedge:	NO	NO	NO	YES
Western Digital WD93028-X is type 3				
Miniscribe model 8225-XT is type 1				

Figure 5

On the Leading Edge

William M. Adney

P.O. Box 531655

Grand Prairie, TX 75053-1655

Copyright © 1991 by William M. Adney. All rights reserved.

Since I originally mentioned QEMM-386 in my column, I have received LOTS of letters about problems in setting it up to work properly on a Z-386. In hindsight, I should have provided a detailed article in setting up QEMM-386 on Z-386 computers, and the purpose of this article is to do just that. To make sure that we all have a common understanding of what is going on, I will "steal" some general information about the Z-386 memory map from one of my previous articles that was published in the February 1990 REMark.

Although I have been using QEMM-386 for quite a while with no problems, I have been amazed at the many problems reported in letters I've received. For those who included a copy of the AUTO-EXEC.BAT and CONFIG.SYS files that I must have to properly diagnose problems, it is obvious that everyone has made the mistake of trying to squeeze every last byte out of the UMBs, and so far as I can tell, that is the major problem.

Although this article is specific to the Z-386 series of computers, the basic information is generally the same for just about any 80386-based system that uses some kind of ISA architecture, which of course excludes the IBM PS/2 series that uses MCA. For those systems, QEMM-386 includes an Adapter Description Library (ADL) which can be used, but that is beyond the scope of this article.

To help eliminate some problems, let's begin by reviewing how the first megabyte of memory is used in a Z-386 system.

The First Megabyte

When I discussed this in the February 1990 article, I did not include hexadecimal

addresses because I thought it would be confusing for many users. Many of you have told me that you are not at all interested in this kind of technical information, but the days when knowing this kind of information was "optional" are rapidly drawing to a close. If you think that you do not need to know this kind of detail, perhaps you have not attempted to troubleshoot memory problems with Windows 3.0. Unfortunately, setting up Windows 3.0 to run properly on a system is not even remotely as simple as

time being.

But when you need to configure or troubleshoot problems with QEMM-386 (or Windows 3.0 for that matter), you will need to know what the hex addresses are in order to use the EXCLUDE command. The hexadecimal numbering system is quite easy to understand for our purposes here.

The hexadecimal numbering system consists of 16 digits: The usual numbers 0-9, plus the DIGITS A, B, C, D, E, and F. To count from 0 to 16 (decimal) in the hex system, the numbers are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. And the conversion is exactly as you might suppose. That is, Ah (hex) is equivalent to 10d (decimal), and Fh (hex) is equivalent to 15d (decimal). Although it's not important to this discussion, I will mention in passing that double-digit and larger hex numbers follow the same rules you are familiar with in the decimal system. For example, the number after Fh is 10h, which is of course equal to 16d.

We talked about Upper Memory Blocks (UMBs) in last month's column, where a memory block was a 64 kilobyte chunk of memory. For technical reasons, it happens that there are 16 sixty-four kilobyte chunks of memory in the first megabyte of address space (for memory). Note that the 16 chunks of memory correspond to the first 16 digits in the hex numbering system. Also, note that, if you multiply 16 times 64 KB, you will get the "magic" number of 1024 KB, which just happens to be one megabyte.

There are several ways that one can talk about memory addresses, but we only need to know about the one used for our purposes here. To keep things straight, we must use 4-digit numbers to indicate memory addresses. Therefore, the first (i.e.,

Setting Up QEMM-386 on a Z-386, CompuTek Systems

many people believe, despite the advertisements that it will simplify the user interface and millions of copies have been sold. I have talked to many PC experts in a number of companies, and they tell me that they will not recommend going to a Windows environment until many of the problems, such as Unrecoverable Application Errors (called UAEs) are significantly reduced or eliminated. In many cases, they have not been able to isolate whether the problem is in Windows or the application. The whole point is that you will also need to know this kind of information to troubleshoot Windows' memory problems, at least for the

FFFF	-----	1024 KB (1MB)
F000	Slushware (Monitor ROM) - 64 K	960 KB
EFFF	-----	
	Slushware (Custom ROM) - 48 K	

	Slushware (EGA Video ROM) -16 K	
E000	-----	896 KB
DFFF	EMS Page - 64 K	
D000	-----	832 KB
CPFF	-----	
	Add-On Cards - 48 K	

	EGA Video ROM - 16 K	
C000	-----	768 KB
BFFF	CGA Video (32 K)	

	MDA Video (32 K)	
B000	-----	704 KB
AFFF	EGA/VGA Video RAM (64 K)	
A000	-----	640 KB
9FFF	-----	
	Conventional System Memory	
0000	-----	0 KB

Figure 1. 1 MB Memory Map for Z-386 Systems

lowest) memory block begins at 0000h, the second memory block begins at 1000h, the third memory block begins at 2000h, and so on. Following this pattern, the tenth memory block begins at 9000h, which is the last memory block in conventional memory. We already knew that 640 KB was the maximum conventional memory size, and that is consistent with our first 10 memory blocks of 64 KB as discussed.

The last six memory blocks continue in the same way, beginning at A000h and proceeding to B000h, C000h, D000h, E000h, and F000h. As you may recall from last month's column, these are the Upper Memory Blocks that are reserved for various purposes. These are the specific memory addresses that we need to know about for correctly configuring QEMM-386. To keep things relatively simple, I will simply note that these 64 KB memory blocks include hex addresses from A000-AFFFh, B000-BFFFh, C000-CFFFh, D000-DFFFh, E000-EFFFh, and F000-FFFFh. Let's continue this by looking at Figure 1, which is a relatively detailed memory map of the first megabyte of memory in the Z-386 series of computers.

The entire box shown in Figure 1 represents the first megabyte of memory on all Z-386 systems, including laptops. The area at the bottom of the box (up to 640 K or 9FFF) represents Conventional System Memory which can be ignored in this discussion. Memory at the bottom of the box is usually referred to as "low memory" because it begins at zero (0000h). Memory closer to the 1 MB address limit (above 640 K) consists of a series of six 64 KB segments or blocks of memory that are called Upper Memory Blocks (UMBs) as discussed in last month's column.

The boundaries for each UMB are represented as horizontal lines showing

how the entire first megabyte of RAM is organized. Conventional System Memory is shown as a total of 640 KB (from 0000h to 9FFFh) rather than 10 blocks of 64 KB.

In case you are unfamiliar with this notation, it is essential to know that one kilobyte (usually abbreviated as K or KB) represents 1,024 bytes, not 1,000. In other words, a system that has 640 K of conventional memory has 655,360 bytes of memory, not 640,000. If you know your system has 640 K of conventional memory installed, you can easily prove this to yourself by running CHKDSK, which will display a message like "655,360 bytes total memory." Even if your system was originally equipped with 1 MB of memory (e.g., any current 80286 or 80386 system), CHKDSK will still display 655,360 bytes of conventional memory because that is the maximum for a PC compatible system, regardless of what CPU the system has.

By definition, a megabyte (usually abbreviated as MB) is 1,024 kilobytes or 1,048,576 bytes (1024 times 1024). Even though the definitions of kilobyte and megabyte are technically accurate, you will still find that some people and documentation define kilobyte as 1,000 bytes and megabyte as 1,000,000 bytes due to sloppiness or misunderstanding.

In any case, 640 kilobytes are used for conventional memory, but how is the rest of the one megabyte address space used? It is important to remember that IBM defined a general standard, or map, of the organization of the remaining 384 KB of memory. The most important point about the remaining 384 KB (the six UMBs) of memory was that it was defined by IBM for a specific purpose and it was RESERVED for that purpose. In this context, RESERVED simply means that this memory was not available to an application program or hard-

ware, other than to be used for its defined purpose. As a side note, several people have asked me why the "Base Memory" on the SETUP screen only shows 640 K when the system is known to have 1 MB of RAM. Perhaps the answer to that question is now clear in that the remaining 384 KB is reserved, although I would agree that a better name for "Base Memory" might be something like "Conventional Memory". That's also the reason that CHKDSK only displays a maximum of 640 KB of memory, too. Now that we have reviewed some of the terminology and the first 640 KB of memory, let's continue with a discussion of how the UMBs are defined.

The Upper Memory Blocks

The first UMB above 640 KB (640 K to 704 K or A000-AFFFh) is reserved for EGA/VGA video RAM. If you have an EGA or a VGA video card in your system, such as the Z-549 that my system has, the UMB is entirely reserved for that purpose. Quarterdeck's Manifest program correctly detects that usage, and my version accurately reports that the A000-AFFFh UMB is used for "VGA Graphics".

The second UMB, between 704 KB and 768 KB (B000-BFFFh), is reserved in two 32 KB pieces. The first (i.e., lowest) 32 KB block from 704 KB to 736 KB (B000-B7FFh) is reserved for the IBM Monochrome Display Adapter (MDA), even though only the first 4 K is actually used (28 K unused). My research indicates that non-IBM monochrome cards may use a larger portion of this 32 KB block, which explains why some software provides special video drivers for these cards, such as Hercules. The block, from 736 KB to 768 KB (B800-BFFFh), is reserved for the Color/Graphics Adapter (CGA), but only the first 16 K is normally used (16 KB unused).

The third memory block, between 768 KB and 832 KB (C000-CFFFh), is reserved for "Miscellaneous Expansion", which can mean all kinds of things depending on what hardware your system has. If your system is like mine with at least one hard drive and an EGA/VGA video display, the installation program will probably take care of that with no problem.

In the original memory definition, the fourth memory block, between 832 KB and 896 KB (D000-DFFFh), was simply reserved by IBM for future use, but it was eventually used on the IBM PC Jr. for ROM cartridges that contained various application programs, such as Lotus 1-2-3. Since that time, that particular UMB has been used by many computer manufacturers on many systems, including the Z-386 series, for EMS paging to handle expanded memory.

The fifth memory block, between 896 KB and 960 KB (E000-EFFFh), was also reserved by IBM for future use. It was also used on the IBM PC Jr. for ROM cartridges,

which explains why that computer had two ROM "slots". This is the point where things begin to get confusing, so we will look at that in more detail in a minute.

The sixth and last memory block, between 960 KB and 1024 KB (F000-FFFFh), was always reserved by IBM for the system ROM (including ROM BASIC). This includes the ROM Monitor, ROM-BIOS, and ROM-BASIC for the IBM computers. The installation program I have seems to correctly detect the usage of that UMB, so that should be no problem. Now let's take a look at how I installed QEMM-386, and then we will make a few changes.

Installing QEMM-386

Although I have tested the recommended QEMM-386 installation and setup on a variety of Z-386 computers, I have obviously done much more extensive testing on my own Z-386/16 system. If you follow these recommendations, you should not have any problem getting QEMM-386 to work correctly on any Z-386 model, but let me stress again that it really depends on what hardware you have installed. You may want to use these recommendations as a starting point to get everything up and running correctly, and then make some modifications to increase the amount of available memory. That's fine, but I would caution you to test ALL changes thoroughly with ALL of your software to make sure there are no problems with the changes you have made. I will give you some clues about how to approach that, but you are on your own if you try anything different from the configuration I have described here.

Before we get into installing QEMM-386, I should include a brief description of my own system which I have used for extensive testing. I have a 16 MHz Z-386 which has ROM version 2.6E (an old ROM) with ZDS MS-DOS 4.0 (actually 4.01 with BIOS 4.00.02 indicated by the VER command). When I use the SETUP command, my system has a Base Memory Size of 640 K, an Expansion Memory of 4096 K, a Video Display indicating "Enhanced Graphics", and a Video Refresh rate of 60 Hz. On the hardware side, I have two memory boards: a 4 MB Z-515 and a 1 MB Z-505, and a Z-549 video card that drives a ZCM-1492 FTM monitor. Note that all memory above 1 MB is configured as EXTENDED memory (i.e., Expansion Memory is 4096 K). For QEMM-386 configuration purposes, I have version 5.11 with Manifest version 1.01, and the rest of the hardware is not important, so I will ignore the hard drives, printers, and some other things.

It is also important to understand what QEMM-386 can do. First, it provides the capability to use available space in the UMBs to load memory-resident programs, device drivers, and DOS resources. As a result, you will have more space available in conventional memory, which can be

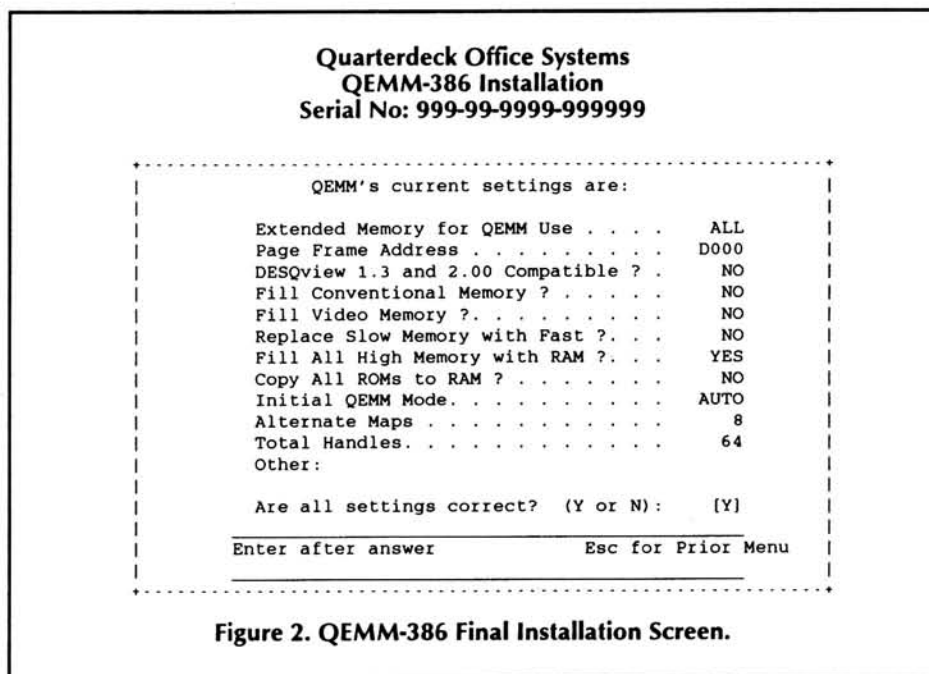


Figure 2. QEMM-386 Final Installation Screen.

critical if you are running a large program. In the extreme case, you may be able to run a large program on a 1 MB system without having to buy any additional memory.

The second major QEMM-386 feature is that it converts extended memory into expanded memory. For systems like my Z-386/16 which have switches on the memory boards that must be changed according to whether you want extended or expanded memory, that is a particular advantage because there is no need to continually reset the switches (which requires disassembling the system and changing the memory configuration in the SETUP command) depending on what kind of memory you want. On later systems, such as the Z-386/25 and Z-386/33, you can easily adjust the memory with the SETUP command, but it is still easier to set everything as extended memory and let QEMM-386 take care of the rest. That is especially true if you need to mix the use of some expanded memory and some extended memory. In short, QEMM-386 has lots of features, and the version 5.11 that I tested is also compatible with Windows 3.0. All of this information and more is included in Chapter 1 (Introduction) of the QEMM-386 manual.

Chapter 2 (Installation) is very important and you should read it completely (pages 7-12 in the version I have) BEFORE you begin installing QEMM-386 because there are two possible changes you MUST make to CONFIG.SYS. First, remove any

expanded memory manager command line. For ZDS MS-DOS 4.0, that specifically means you must delete a line that contains a DEVICE= command which includes either EMM.SYS or EMM386.SYS. Depending on your system, you will probably find that EMM386.SYS was installed if you are running Windows 3.0. Second, you must also delete the line that contains a DEVICE= command that specifies HIMEM.SYS, which is usually added by the Windows 3.0 installation program. And finally, be sure to read and understand the section on "Installing QEMM-386 with Windows 3.0" if you have already installed Windows. I have worked with this version of QEMM-386 for some time, and the instructions are quite good. Just be sure you follow them exactly, especially if you have installed Windows 3.0.

The QEMM-386 INSTALL program is straightforward, and I will assume that you can go through the initial screens with no problem. Generally, the program will ask you for your name and the subdirectory you want to use for the software. For QEMM-386 version 5.11, the program will present a screen that looks something like Figure 2.

NOTE THAT THE ANSWERS SHOWN IN FIGURE 2 ARE NOT THE DEFAULTS PROVIDED BY QEMM-386. You must FIRST answer N to the "Are all settings correct?" question and answer individual questions like those in Figure 2. When you have answered the individual questions as

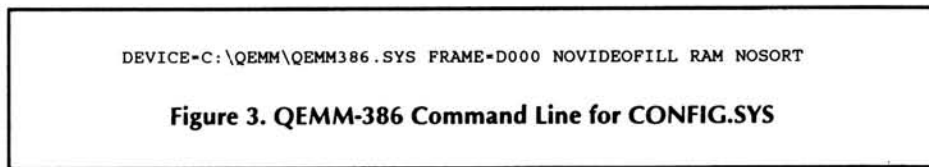


Figure 3. QEMM-386 Command Line for CONFIG.SYS

```

DEVICE=C:\QEMM\QEMM386.SYS FRAME=D000 I=E000-EFFF NOVIDEOPILL RAM NOSORT
rem Include the E000 segment and change it if problems occur
INSTALL=C:\QEMM\LOADHI.COM /TSR /R:1 C:\DOS\SHARE.EXE
SHELL=C:\COMMAND.COM/E:256/P
rem Add /X to BUFFERS per QEMM-386 suggestion for DOS 4.0
BUFFERS=35 /X
rem Change FILES to 10 (from 40); make change to AUTOEXEC.BAT (QEMM-396)
FILES=10
BREAK=ON
rem Add LASTDRIVE per QEMM-386 suggestion
LASTDRIVE=D
rem Ignore QEMM-386 suggestion on STACKS=0,0
STACKS=16,128

```

Figure 4. Final CONFIG.SYS File Example.

recommended, your FINAL installation screen should look like Figure 2. Then the program will create a command line for your CONFIG.SYS file which should look something like Figure 3.

The only thing that is really unique about the command line in Figure 3 is that the EMS Page Frame is explicitly stated (FRAME=D000). The whole reason for that is because I found that all of the ZDS technical manuals I have stated that D000 segment is designed to be used for EMS paging. Since I found that the QEMM-386 program automatically used the E000 segment if I did not specify it, I decided to configure the program to be consistent with the ZDS design intentions. Also, note that the command line shown gives QEMM-386 complete control of all 4 MB of my extended memory and converts it to expanded memory. And since I occasionally need 1 MB extended memory for testing, I simply add a MEMORY=3072 command to Figure 3, which gives QEMM-386 control of 3 MB as expanded memory and the remaining 1 MB stays as extended memory. As you can see, this process is really quite easy so long as you follow these steps.

After the installation is complete, you can let the installation program reboot your system which is required to install the new command line that has been added to CONFIG.SYS. Also be sure to at least briefly look through the READ.ME file to see if there is anything new you should know about. My personal preference is to print it for review, and then I keep it in a file folder so I can delete that file from my hard drive.

The Next Step

The next step is to run the OPTIMIZE program. That program checks to see what device drivers, DOS resources, and memory-resident programs can be loaded into available UMB space, which will increase the amount of conventional memory available in your system. Before you run OPTIMIZE, I recommend you run CHKDSK and record the amount of "free memory" so that you will be able to compare it later.

After you run OPTIMIZE, you will probably find that the LOADHI program is used to load various programs into the "open" space in various UMBs. After running OP-

TIMIZE, you should also refer to Chapter 6 (The DOS Resource Programs) to help with additional changes that will save memory. Again, it is very important to read the manual because you may find something that is VERY important to your system. For example, there is a special NOTE in the BUFFERS.COM section of my manual which clearly states that the program should not be run in DOS 4.0 systems because the structure of the disk buffers is different. Of course I did not use that program because I am running version 4.0.

I did use FILES.COM, FCBS.COM, and LASTDRIV.COM as well as Manifest to see what was going on in my system. After I finished with all of that, my CONFIG.SYS file looked like Figure 4.

I also made a few changes to my AUTOEXEC.BAT file as a result of using these programs, and the final result is shown in Figure 5.

NOTE that the examples shown in Figures 4 and 5 are EXAMPLES ONLY, and YOU SHOULD NOT COPY THEM for your system. I was hesitant about including an exact listing of the my own files because I always seem to get at least one letter from someone who has copied exactly what I personally use in my system. And of course my files generally will not work in other systems unless you have exactly the same configuration that I do. But I always get requests for a working example of what I use, so I decided to include the listings. And in case you are wondering, both of the programs in the \ADNEY subdirectory are

utility programs that I wrote in Turbo C.

So much for the general configuration. But there are some additional details that I found in the Technical Manuals on the Z-386 series which are worth looking at.

Slushware

For a computer to be compatible, it must generally use the kind of memory map shown in Figure 1. The Z-386 series has a feature called SLUSHWARE, which is not exactly hardware; it really isn't firmware, and it is not altogether software. SLUSHWARE is program code (software) that is copied from the system ROM (firmware) into RAM (hardware). Because RAM is much faster than ROM, system performance is significantly improved. And although I used the ZDS Technical Manuals when I developed Figure 1, I found that not everything turned out the way I expected when I used Manifest and QEMM.COM to analyze my system. As a result, I made some decisions that seem to contrary to what the Technical Manuals state, and although I have not had any problems with any software that I use, I think it is important to review those decisions. I think it is important that you understand what I did, just in case the way I implemented QEMM-386 does not seem to work on your system.

As shown in Figure 1, the Z-386 Technical Manuals clearly state that the entire UMB beginning at F000h is reserved for the Monitor ROM. Since Manifest displays the fact that the segment has been associated with ROM, I did not find any problems. But, I have received a lot of letters from people who have tried to INCLUDE pieces of that segment as RAM. These same people have asked me why a printer, keyboard or something else does not work. The answer is that that part of the Z-386 ROM slushware has been clobbered by an attempt to use it as QEMM-386 RAM. As previously stated, that segment has been totally RESERVED on the Z-386 series, and all bets are off if you try to use any part of that segment as RAM. If you think you really need to add something, then I suggest adding the AROM=F000-FFFF command which will completely exclude that

```

@ECHO OFF
rem autoexec.gem
PROMPT $P $Q$Q$G
PATH C:\PF6;C:\;C:\DOS;C:\BATCH;C:\PUP;C:\UTIL;C:\HUG;C:\PCT
PATH %PATH%;C:\PRO;C:\SB5;C:\QEMM
rem Add FILES command per QEMM-386
C:\QEMM\LOADHI FILES=40
rem Set screen blank for 10 mins with old z-449 driver
C:\QEMM\LOADHI /R:1 C:\UTIL\SETVID SCRNSAV 10
rem Set keyboard repeat rate
C:\ADNEY\FASTKEY 12
rem Unlock Num Lock
C:\ADNEY\UNLOCK
ECHO ON

```

Figure 5. Final AUTOEXEC.BAT File Example.

UMB from QEMM-386. Even if you think you have found some "holes" in that segment (as reported by Manifest, for example), don't even attempt to use that space because you may eventually find something that does not work correctly six months or a year later.

The next UMB, beginning at E000h, is another story. As shown in Figure 1, the bottom 16 K, beginning at E000h, is used for EGA Video ROM slushware, which leaves the remaining 48 K for Custom ROM Slushware, which is documented in some parts of some Z-386 Technical Manuals. Other parts of the same Technical Manuals say that the entire 64 K may be entirely allocated to "optional" (i.e., custom) ROM code or 16 K may be reserved for EGA ROM code. So much for what the manuals say.

I did some extensive research on the contents of the various UMBS, both before and after the installation of QEMM-386. I found some things that were quite surprising to me. I used DEBUG to examine the contents of both the C000h and E000h segments BEFORE I installed QEMM-386. After starting DEBUG, I used the DC000:0000 command and the DE000:0000 command to Dump the first 128 bytes of each segment. I found the Z-549 video card signature in BOTH segments, and after comparing the contents of each, I found that they were pretty much the same. Although I am still not absolutely certain, it seemed like the differences could be accounted for by the fact that the C000h segment did in fact contain the EGA Video ROM code as expected, but the E000h segment was EGA Video ROM SLUSHWARE. It also happens that both QEMM-386 and Manifest do NOT show any usage on that segment, and it was automatically mapped as RAM. More importantly, I found there was no problem in using the E000h segment for additional memory, even though I expected to find that my video display slowed down a little because I prevented EGA Video slushing mentioned in the manuals.

Perhaps you see why I added the I=E000-EFFF command as shown in Figure 4. If I find any problems in the future, I can simply change the I for Include to X for eXclude to see if that makes a difference. If you find any kind of strange problem, I suggest you begin with that change. So far as I can tell however, my decision to Include that segment will only disable the EGA ROM slushing, but it does not have any effect on my video display. In any case, it has worked fine for a number of months, so I doubt you will have any problem with it, even if you are using a ZDS video card. By the way, I should mention that I have heard some other brands (e.g., Paradise) of video cards support ROM slushing, so you should be sure to check the video card manual for non-ZDS products. That brings

up another VERY important point that is critical to the proper installation and use of QEMM-386, regardless of what computer you are using.

A Caution!

By its very nature, QEMM-386 is EXTREMELY HARDWARE DEPENDENT, especially within the first megabyte of RAM. Much of the confusion surrounding the installation of this product seems to be that this fact is not recognized and clearly understood. For example, I received one letter stating that the user had upgraded to a new ROM version on a Z-386/16, and he stated that the new ZDS ROM was obviously not compatible because QEMM-386 would not work. Not True! It turned out that he had mapped some space in the F000h segment (FORBIDDEN!) as RAM, and apparently the ROM code that was supposed to be there in the new version got clobbered. You should double check for changes any time you install new hardware, which may include a video board or a hard drive (or anything else), because different manufacturers use the address space between 640 KB and 1024 KB in different ways.

The other major mistake I've seen is that a number of people are trying to squeeze every last byte out of high memory without thorough testing. Although this QEMM-386 and Manifest software is really outstanding and highly recommended, it is not perfect (but is close to it). It does however, provide all the tools you need to find problems, assuming you are will to spend the time it takes to use them. Manifest and QEMM.COM can help significantly, but it will take a lot of time to test EVERY change for an Include statement you add. For example, you may not find a

problem because you failed to test a program's graphics mode (in Word Perfect 5.1 for example) until it actually occurs. And each piece of application software is different, so be sure you test everything that you use to be sure you won't have an unexpected problem at the worst possible time.

Speaking of software, you will probably be well advised to NOT use QEMM-386 when you are running games, especially copy-protected ones. Some of the copy protection schemes do really strange things to UMBS, so you may want to keep that in mind as a possibility.

Although this version of QEMM-386 is compatible with Windows 3.0, the configuration I have discussed in this article is NOT designed for use with Windows. In fact, I have not even attempted to configure QEMM-386 for use with Windows because I just don't use Windows much, at least not yet. As I have mentioned in previous articles, I use DESQview for multitasking because it is so much faster and does not require a major investment in new software to use it.

Thanks for the Memory

Now that you have seen my QEMM-386 configuration, it may be helpful to see a quick overview of how the first megabyte of memory in my system is allocated. I used Manifest a lot to see what was being used and how. Figure 6 was obtained from that program using the Q (QEMM-386), Y (Type) command.

Perhaps you will find the information shown in Figure 6 useful in setting up QEMM-386 on your system.

Computek Systems

As I was looking through my July 1991

Overview	Type	Accessed	Analysis	Memory
n=0123	4567	89AB	CDEF	+ = Mappable
0n00	XXXX	XXXX	XXXX	* = Rammable
1n00	++++	++++	++++	F = Page Frame
2n00	++++	++++	++++	H = High RAM
3n00	++++	++++	++++	M = Mapped ROM
4n00	++++	++++	++++	X = Excluded
5n00	++++	++++	++++	V = Video
6n00	++++	++++	++++	A = Adapter RAM
7n00	++++	++++	++++	R = ROM
8n00	++++	++++	++++	/ = Split ROM
9n00	++++	++++	++++	
An00	VVVV	VVVV	VVVV	
Bn00	HHHH	HHHH	VVVV	
Cn00	MMMM	MMMM	HHHH	
Dn00	FFFF	FFFF	FFFF	
En00	HHHH	HHHH	HHHH	Press F3 for
Fn00	RRRR	RRRR	RRRR	List Mode

Figure 6. Manifest (Q)EMM-386 T(y)pe Display

Quality Enhancements!

EaZy PC Products

EZM-128: Expand 512K base memory to 640K. Simple, plug-in installation. \$95.00

EZCLOCK: Calendar/Clock. Piggy-back add-on for EZM-128. \$35.00

No Slot Clock/Calendar

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

H/Z-148 Expansions

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

ZP-148: PAL chip expands existing 640K memory to 704K. CGA/MDA only! \$19.95

Configuration Control

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

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

VCE-150: Eliminate video card. Install EGA or VGA card. All plug in. Includes circuit board, SRAM and RM-150. \$49.95

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

ULTRA-PAL: Three PAL chips: MR150 for 704K + 512K RAM Disk; MR150T for 640K + 512K RAM Disk; LIM150 for 640K + 512K (32 pages) of simulated v3.2 Lotus/Intel/Microsoft Expanded Memory. With software. Install on standard memory card. No soldering. Needs 45 256K RAM chips (not included) for maximum memory configuration. \$39.95

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

H/Z-100 Modifications

ZMF100A: Expand "old" motherboard (p/n 181-4917 or less) using 256K RAM chips (not included). No soldering. \$65.00

ZRAM-205: Put 256K RAM chips on your Z-205 board. Get 256K plus 768K RAM disk. Contact us for data sheet before ordering. Without RAM chips. \$39.00

H/Z-89 Add-Ons

H89PIP: Parallel printer 2 port interface card. With software. \$50.00 Cable \$24.00

SLOT4: Add fourth expansion slot to right-side accessory bus. \$39.95

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

FBE

FBE Research Company, Inc.

P.O. Box 68234, Seattle, WA 98168

206-246-9815 Voice/FAX TouchTone Selectable

article for something else, I noticed that I had forgotten to include the address and phone numbers for Computek Systems. Since I did note that it was located in Dallas, I suspect that a number of you may have called directory information to obtain the phone number. And since I also mentioned that Computek Systems is an authorized ZDS Service Center, some of you may have noticed the toll-free number that appears in the REMark masthead: (800) 777-4630. Or you may have written to me. Or you could have waited until I noticed it was missing from that article and mentioned it again. To all of you, my apologies, especially to Chris Ziem, owner of Computek Systems. It is included at the end of THIS article.

Powering Down

I hope you will find that QEMM-386 is as useful as I have. And in case you want a comparison, I now have 592,386 bytes of memory free as reported by CHKDSK. That recovered about 28 K of memory that I can use for other things, not to mention DESQview.

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

to an existing board or on any add-on board), all other internal add-on boards (e.g., modem, bus mouse or video card), the brand and model of the CRT monitor you have, and the brand and model of the printer with the type of interface (i.e., serial or parallel) you are using. Also be sure to include a listing of the contents of the AUTOEXEC.BAT and CONFIG.SYS files unless you have thoroughly checked them out for potential problems (e.g. TSR conflicts). If the problem involves any application software, be sure to include the name and version number of the program you are running when the problem appears.

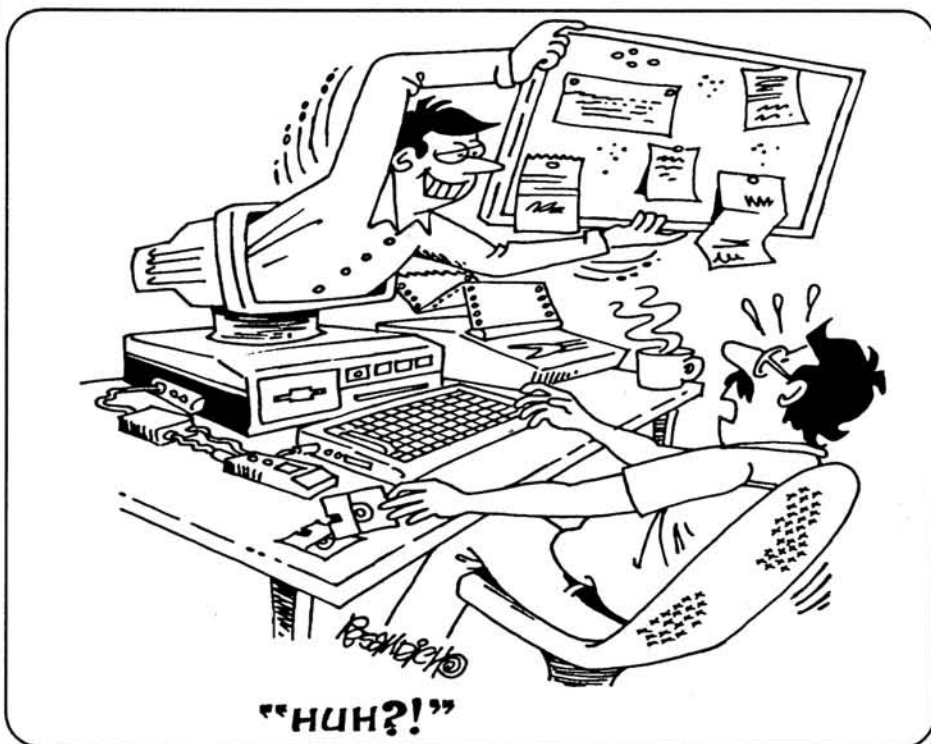
If you have questions about anything in this column, or about Zenith Data Systems or Heath computers in general, be sure to include a self-addressed, stamped envelope (business size preferred) if you would like a personal reply to your question, suggestion, comment or request.

Products Discussed

Computek Systems
10703 Plano Road, Suite 300
Dallas, TX 75238
Phone: (214) 503-6500
FAX: (214) 340-9482

Software

Manifest	\$60.00
GRAM	80.00
QEMM386	100.00
DESQview	130.00
DESQview386	220.00
Quarterdeck Office Systems	
150 Pico Boulevard	
Santa Monica, CA 90405	
(213) 392-9701	



External Portable Hard Drive for Your Laptop or PC-Compatible

(The ex-PORT Hard Drive by Lightek)

Henry E. Fale
Quikdata, Inc.
2618 Penn Circle
Sheboygan, WI 53081

Are you one of those who purchased a laptop without a hard drive? Do you have a MinisPort and have wanted hard drive capabilities? Or perhaps you would like an easy and convenient way to transfer files from one system to another. If you have a PC compatible with a parallel port, especially a laptop, and you want to add a hard disk to the unit, then this article is for you.

We have recently evaluated the UCI/Lightek ex-PORT external 20 megabyte hard drive unit. This is a compact unit powered by 110-volt house current, which connects to your computer via a parallel port. Thus, it will work with any laptop and any computer that has a parallel port, which is all of them. No cards to install, no complicated interfacing, and no complex software installation required. Easy and simple.

I receive many calls asking if a hard drive is available to those who have laptops that did not have a hard drive built in. For a while, ZDS had a unit which would attach to certain laptops with an expansion port built in. The problem was it cost about \$1000 for a 20 megabyte unit, and only worked with a select few laptops. This unit is under \$500 and will work with anything.

This portable unit measures 3" wide, almost 6" tall and about 16" deep. On the front panel is an off/on switch, a red LED for power indication and a green LED for drive access indication. On the rear is a small jack for the power cube transformer which is included, and two DB25 connectors; one for the parallel interface from the computer's "printer" connector and another for a device (printer) that you had or may have plugged into the computer's parallel port connector. So even when this is being used, you will not lose your parallel port. Thus, the unit is quite compact and simple in design.

Inside the unit is a hard drive, and an

AT bus connector. A standard AT (16-bit) hard disk/floppy disk controller is plugged into this bus connector, and also the Lightek ex-PORT interface card.

Since this unit is actually a 16-bit miniature AT motherboard, it's much faster than some of the 8-bit expansion chassis that we find around. The hard drive controller is actually capable of controlling two floppy drives and two hard drives. In fact, UCI makes several different units, with some having several expansion slots and places for several floppy and hard disks (they are still working on software for the floppy units). I understand that Lightek has even tried this unit using Novel Ethernet, several different hard drive controllers (IDE, ESDI, RLL and MFM), 360K floppy drives and 1.2 meg floppy drives (for AT computers only). They have also tried a tape drive with controller and tape software and a Mitsubishi scanner. But I won't get into all of that in this article. I am reviewing the hard drive part.

The first thing we had to do was convert the software. Most folks will probably use something like this with a laptop computer, so the supplied 3.5" disk should not pose any problems. Since I wanted to try it on various machines, I had to convert the 35" supplied software to both 5" 360K and 2" MinisPort formats. That way, I could try the unit on everything — and I did. When we test something, we test it under all possible conditions. I don't want to sell anything that's going to cause more support headaches than it should. I spend far too much time on support the way it is. If a product gives too much trouble or does not work as specified, or on a range of machines it is meant for, back it goes.

I have currently tested this unit on a Zenith Data Systems Z-151 computer, a Z-248 AT computer, a Z-159 XT computer, a

MinisPort computer and a Sanyo XT clone computer. It performed lawlessly in each case. In all my tests, it appeared that the ex-PORT was just about as fast, or as fast, as the built-in hard drive, if the machine had one, and much faster than the machine's floppy, in the case of the MinisPort. This is surprising since all the information must be sent to this unit through the computer's parallel port. But it does work. The marvels of computers and the human mind never cease to amaze me. I mean, who ever thought that you could work an external hard drive through a parallel port!

This unit is especially suited for laptop computers and it will do justice in these cases. If you were one of the unfortunate who purchased a MinisPort before the 20MB hard drive units became available, or purchased a laptop without a hard drive, now is your chance to catch up — and at a price you can afford. The only possible drawback is that it requires house voltage to work, so it cannot be used in portable applications. I consider it to be portable because it can easily be carried around and even connected easily from machine to machine. I've already got a good use for mine. I can connect it to my MinisPort and have all those files where I want them and when I want them. The light weight MinisPort can still sit on my lap, and the hard drive can be on a nearby table. Quite convenient. Then when I'm finished, I simply connect it back to my main computer and load the files back. You could, of course, do the same to transfer files between two desktop computers. And for a 20 meg. unit for under \$500, the price is right.

To use this unit, you simply plug in the power cube to an AC outlet, connect the small power connector to the ex-PORT, and attach the parallel cable from the com-

puter printer port (LPT1: or LPT2:) to the ex-PORT. If you are using a printer, simply plug it into the rear of the ex-PORT. Now load in the two driver files, and add them to your CONFIG.SYS, and you're ready to go. Turn the unit on about 15 seconds before the computer boots, so when CONFIG.SYS comes around it will "see" the ex-PORT. When the computer boots up it will assign the ex-PORT hard drive to the next drive in the system, and from that point on, you simply use it as a system drive.

The format for the CONFIG.SYS is the addition of these files:

DEVICE=EXPORTAT.EXE /W0
DEVICE=EXPHDESD.EXE

That's all there is to it. The device drivers in the CONFIG.SYS file communicate with your computer to the ex-PORT unit. The driver handles the interfacing with the driver card inside the ex-PORT expansion chassis. The needed information passes through the parallel port to the ex-PORT, being handled properly by the device driver. They even supply, in the Appendix level programming interfacing routines for accessing ex-PORT device driver routines. Thus, anyone familiar with Assembly Language programming can adapt other de-

vices and software to the ex-PORT.

The EXPORTAT.EXE driver contains support for DMA (Direct Memory Access) mode in addition to the timer activated DMA access mode. I suppose DMA is one reason it is so fast. It also has auto-detection of the parallel port, LPT1:, or LPT2:, so you normally don't have to mess with that. The /w0 switch means no wait states, and is the default. This value can be up to 9, but in every test I ran and with every computer, I never had to touch this value. I understand that for some fast 386 systems you may have to make an adjustment.

The EXPHDESD.EXE contains the AT hard drive driver used with all MFM, RLL or IDE 16-bit hard disk controllers. It supports 1:1 interleave, which is another reason it is so fast.

In addition to these files, the distribution disk also has files to low-level format the hard disk and floppy drivers, partitioning software, 8-bit hard disk controller driver, object code file for Novell Netware NE1000 network card, assembly code for interfacing with the ex-PORT device driver, sample CONFIG.SYS file, and more. Normally, the only files needed will be the first two men-

tioned for CONFIG.SYS unless you must low-level format the thing. Units purchased from Quikdata will already have that done.

I won't go into anything more about the additional files, they are covered in both the instruction manual and in a README.DOC file on the disk. And that's about all to say for this unit. It's easy. Plug it all in, configure the software, turn it on, and use it.

Units come in 2-, 3-, 5-, and 8-slot versions. One slot is always taken by the interface card, so you have 5-1 left for your applications. That's why the ex-PORT unit for the hard driver has two slots. I will only say about the others that as they get larger, the power supply also increases. For the larger units, a fan is included and the power supply is built-in. They also have bays for tape drives, hard disks, floppy drives, etc. The unit comes with a 1 year warranty. And yes, the drives are autopark, so when you're done, just turn it off and carry it away.

Approximate prices are as follows: The 20 megabyte unit goes for about \$495, 40MB unit will sell for about \$625, and the 80MB unit will sell for about \$795. ✱

W S Electronics

(513) 376-4348

**** Since 1975 ****

(513) 427-0287

1106 State Route 380, Xenia, Ohio 45385

W S ELECTRONICS INTRODUCES KRIS TECHNOLOGIES

System 48C-3
80486-33MHz CPU
and
System 48C-2
80486-25MHz CPU

- Internal co-processor and 8KB cache.
- 128K secondary cache onboard.
- Socket for optional Weitek co-processor.
- 2MB 70ns RAM, expandable to 16MB onboard.
- Shadow RAM for System & Video BIOS.
- ROM-based setup utility.
- Six 5.25" drive bays available.
- Eight 16-bit expansion slots.
- Dimensions: 8.5"Wx18.5"Hx18"D.

\$3550⁰⁰
and
\$2899⁰⁰

System 38SX-2
386SX-20MHz CPU
and
System 38SX-1
386SX-16MHz CPU

- Socket for 80387SX co-processor.
- 1MB 80ns RAM, expandable to 8MB onboard.
- Page mode interleaved memory design.
- Shadow RAM for System & Video BIOS.
- Two 5.25" & two 3.5" drive bays.
- Six 16-bit & two 8-bit expansion slots.
- Dimensions: 7"Wx13"Hx16.5"D.

\$1225⁰⁰
and
\$1025⁰⁰

System 38M
80386-33MHz CPU
64K cache

- Socket for 80387 or Weitek co-processor.
- 2MB 70ns RAM, expandable to 16MB.
- Page mode interleaved memory design.
- Shadow RAM for System & Video BIOS.
- ROM-based setup utility.
- Two 5.25" & two 3.5" drive bays.
- Six 16-bit & two 8-bit expansion slots.
- Dimensions: 7"Wx13"Hx16.5"D.

\$1999⁰⁰

System 28
80286 60-bit CPU

- Socket for optional 80287 co-processor.
- Standard 1MB RAM onboard, configurable to 640K base & 384K extended memory.
- Three 5.25" & one 3.5" drive bays.
- Six 16-bit & two 8-bit expansion slots.
- User selectable 8/4/2 DMA clock speed.
- Dimensions: 17"Wx6.5"Hx16.5"D.

\$750⁰⁰

System 38
80386-35MHz CPU

- Socket for 80387 or 80287 co-processor.
- 2MB 80ns RAM, expandable to 16MB.
- Page mode interleaved memory design.
- Shadow RAM for System & Video BIOS.
- Four 5.25" & two 3.5" drive bays.
- Six 16-bit & two 8-bit expansion slots.
- Dimensions: 19"Wx6.5"Hx16.5"D.

\$1499⁰⁰



*ALL COMPUTERS HAVE A 2 YEAR PARTS, 18 MONTH LABOR WARRANTY!
*Monitors, Video Cards, Harddrives are optional.

Expanded Memory

What is it All About?

John Eichenberger
1718 Thayer Drive
St. Joseph, MO 64508

If I had to pick one computer-related subject which I have heard far too many questions about it would have to be questions about expanded memory. Normally, the people asking the questions haven't the slightest notion what makes expanded memory different than extended memory. What with so many other "types of memory" and new ones still being introduced, the problem is getting worse instead of better. I can't explain everything in one article (even if I knew everything), so I have chosen to concentrate on writing about expanded memory. Here is a list of some of the questions which you should be able to answer after reading and understanding all of this article:

1. What are the differences between conventional memory, expanded memory, extended memory, and backfill memory?
2. What do the acronyms LIM, EMS, EMM, and OS/E stand for?
3. What are the differences between a version 3.2 EMM and a version 4.0 EMM?
4. What are the EMS 4.0 hardware features, and which ones are typically implemented by any particular EMM?
5. What are the differences between hardware EMMs and software EMMs?

What is the Difference Between Memory and Memory Addresses?

Before I begin I should point out that I will writing about Random Access Memory (RAM). That is to say memory which can be both read and modified. Memory which cannot be modified is typically referred to as Read Only Memory (ROM). For simplicity I will refer to RAM simply as memory.

Memory is accessible because it is "mapped" to a memory address. That is to say there is a memory address which can be used to access the memory. Memory which is mapped to a memory address is said to be addressable. The smallest addressable unit of memory is a single byte. A byte of memory is a number which can take on 256 different values.

If you think of a byte of memory as being a person and a phone system as the only means used to communicate to people, the phone number for each person would be the "memory address" for that person. Furthermore, a person without a phone number would be considered inaccessible. Not non-existent mind you, just unaddressable. If and when a phone number is assigned, the person becomes addressable. Using this analogy, each phone number is a memory address. The number of people who can be addressed is limited to the number of phone numbers available. For example, within a single business typically four digits are used, thus only 10,000 people could be addressable at any one time.

Memory addresses are similar to phone numbers except that they are usually written as hexadecimal numbers. There are two forms for writing a memory address. The form which I will consistently use what is known as a linear address. Linear addresses are written using anywhere from four to eight hexadecimal digits. Hexadecimal digits are '0' through '9' and 'A' through 'F'. In the original PCs all memory addresses could be specified using five such hexadecimal digits. That is because 8088 based systems can only address 1M

(1,048,576 bytes) of address space. On the other hand, 80386 DX machines can address a full 4G (4,048 megabytes) of address space. Thus, on 80386 DX machines you have to use a full eight digits to address any particular byte of memory.

What is Conventional Memory?

In the original PCs, 128K (131,072 bytes) of memory was considered to be quite a lot of memory. That is partially what lead to the design which now restricts MS-DOS to 640K of contiguous memory. In PCs based on 8088 microprocessors, there are only 1M memory addresses. Thus, even if memory was mapped to every such address, the system would be limited to 1M of memory. However, in order to make the machine work some of those addresses have to be reserved for system code and data, some are reserved just in case you want to add a hardware adapter that needs some address space (for whatever reason), and some have to be reserved for memory associated with the video adapter you are using. The start for these reserved regions is where the 640K barrier comes in. In a PC with 640 kilobytes (640K) of memory, the address of the last byte of conventional memory is 9FFFF. The next address, A0000h, is reserved for high resolution video adapters. This memory between address 00000 and A0000 is known as conventional memory. Conventional memory is always mapped, that is to say, it is always accessible. There is always one and only one memory address which can be used to get to any given byte of conventional memory.

What is Expanded Memory?

Well, as we all now know 640K of memory is simply not enough. The natural solution this problem is to add more memory. The problem was, and still is, where? All available memory addresses (on an 8088 based machines) were already spoken for. When the crunch became severe enough, it was decided that sometimes it was more important to use some of the reserved address space for memory rather than reserve it for some other sort of add-on hardware. Even still, there was not much reserved address space left available. Somehow there had to be a way to pack a whole lot of memory into a very small address space. Expanded memory does just that.

Expanded memory uses a small block of address space to access a large amount of memory. The way it is done is most easily explained by returning to the phone system analogy described before. To communicate with a large number of people using just a few phone numbers we could periodically reassign phone numbers so that the person we want to talk to has a phone number. In fact, any time the person we want to talk to does not have a phone number, just give him/her a phone number which is not currently in use. Thus, if we had 1,000 people we could talk to any four of them simultaneously using only four phone numbers. Of course, switching phone numbers all the time is kind of tedious, but then that's where computers come in. Computers slow down a little when things get tedious, but they don't complain.

To make the analogy more accurate, let's say we have 256K (262,144) people. Any 64K of those people have phone numbers at one time. Those phone numbers can be reassigned in blocks of 16K. If at any time we want to talk to a person who does not have a phone number, we have to reassign phone numbers for 16K people and in the process give a phone number to the person we want to talk to. Of course the main reason this analogy is no good is that usually memory is accessed in large quantities. For example, it is quite normal for a program to "talk" to 10,000 bytes of memory at a time but it is quite rare that a person would talk to that many people at time. Having 256K of memory and being able to talk to any 64K of it at a time is a fairly typical example of how expanded memory works. The 16K blocks are referred to as pages. 16K of address space (phone numbers) is referred to as a "physical page". 16K of actual memory is referred to as a "logical page". This seems backwards to me, but I didn't define the terminology. Changing which logical page is associated with a physical page is called either "paging" or "mapping" memory.

When expanded memory was first developed, the applications which made use of expanded memory had to manipulate expanded memory hardware directly in order to use expanded memory. It soon became obvious that this was not the ideal way to go. That's why Lotus, Intel, and Microsoft developed and made public an Expanded Memory Specification (EMS). In the 3.2 version of the EMS document, expanded memory hardware is defined as an abstract item. It is no longer possible for an application to find out exactly how to access expanded memory hardware directly. It is now the sole responsibility of an Expanded Memory Manager (EMM) to manage the hardware. The EMM provides a set of function calls which applications use to determine how much memory is available and to perform operations like mapping a page of expanded memory. Ultimately, what the EMM provides is an interface which makes it possible for multiple applications to access expanded memory without any conflicts occurring. By hiding the hardware specific information, the application need know very little about how the EMS hardware actually works. In fact, the EMM may actually supply expanded memory using many different types of EMS hardware without the application being aware of such details. This is the *real* advantage to having an EMM manage the hardware.

Formal Specifications

Because Lotus, Intel, and Microsoft are the founders of the EMS document, one common term for this type of memory is LIM memory. The other, more common name, is EMS memory or even LIM EMS

memory. There are currently two well known versions of the EMS document. Version 3.2 is essentially the first major publication of the specification. The current specification is version 4.0. Don't confuse version 4.0 of the EMS document with version 4.0 of MS-DOS. The version number is specific to the EMS document, not the version of MS-DOS used. The document number for the revision I have is 300275-005 copyrighted by Lotus Development Corporation, Intel Corporation, and Microsoft Corporation in October, 1987. The specification has been released into the public domain. A copy can be obtained from Microsoft. I have also uploaded a machine readable copy of the EMS document to ZLink-COM1. The EMS document describes in fairly good detail all of the information required to write an application which uses expanded memory. I don't intend to recreate all of the information in that specification, but I will summarize some of it. I will start with how the version 3.2 specification since it is much simpler than the 4.0 specification.

First and foremost, applications programmers have to understand where expanded memory is located. Expanded memory is not assigned an address until the EMM assigns one. In version 3.2, a single set of four addresses are the only addresses to which a logical page may be mapped. The four addresses must be adjacent and start at 16K memory addresses; such as D0000, D4000, D8000, and DC000 as shown in Figure 1. One way to remember this rule is that the last three digits of the starting for a physical page must be zero and the previous digit must be a '0', '4', '8', or 'C'. The address for the first such page is

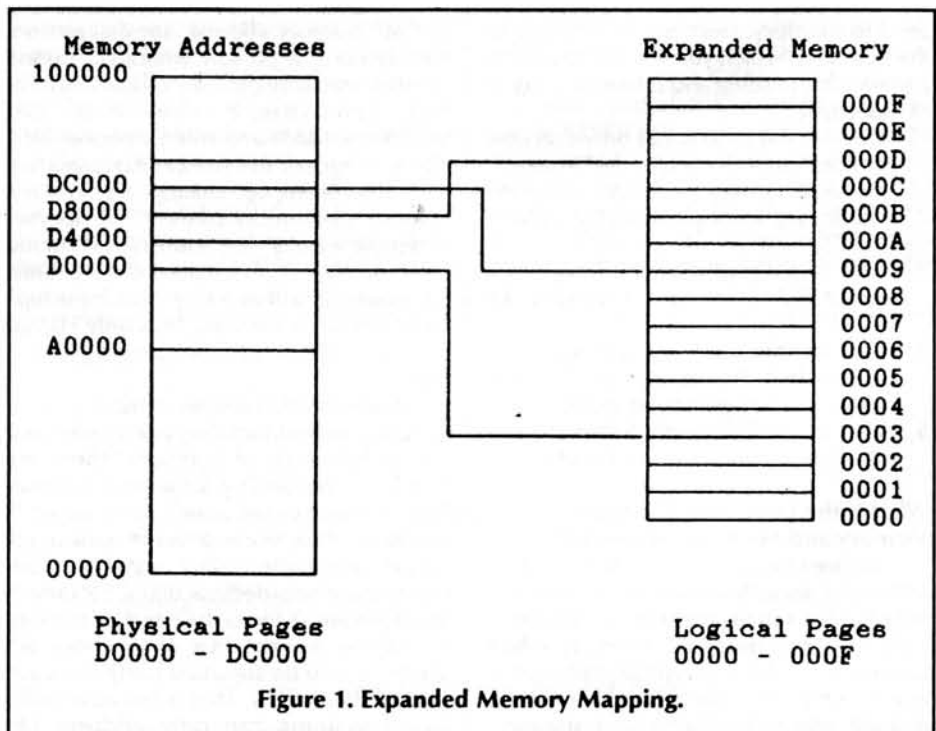


Figure 1. Expanded Memory Mapping.

What is an EMM?

typically a configuration option which depends entirely upon your hardware and software. As a set, the four physical pages are referred to as the "page frame".

On the left side of Figure 1 is a graphic illustration of the first megabyte of address space in a personal computer. The four pages starting with address D0000 represent the expanded memory page frame. On the right is a graphic representation of 256K of expanded memory (note that this figure is not to scale). Lines are drawn to indicate which logical pages have been mapped to a physical page. You will notice that I have numbered the expanded memory on the far right-hand side. These numbers need a name. The EMS document refers to these numbers as logical page numbers. Actually, it uses the term "logical pages" in two slightly different ways which I'd rather not explain just yet. In Figure 1, logical page number 0003 is shown as being mapped to address D0000. That is the way of referring to the fact that address D0000 has been assigned to logical page number 0003. Likewise, logical page 0009 is mapped to DC000 and logical page 000D is mapped to D8000. It is also possible to map one logical page to two physical pages simultaneously. When this is done, the memory can be modified at one address and the modifications will also show up at the other address. The converse is not possible, two logical pages may not be mapped to the same physical page.

What is Version LIM EMS 3.2?

Ok, I have defined physical pages and logical pages. Now, what can we do with them? Well, let me summarize what functions are defined in the LIM EMS 3.2 document. By function number, the functions defined are:

1. Get Status: This function is nearly useless, it just lets you know that the EMM is still working.
2. Get Page Frame Segment Address: Returns the starting address for the page frame.
3. Get Unallocated Page Count: Returns how many logical pages are present together with how many are available.
4. Allocate Pages: Reserves logical pages for a specific application. A handle is returned if the allocation succeeds. The handle is used to distinguish what memory each application owns.
5. Map/Unmap Handle Pages: Causes a single logical page to be mapped to a particular physical page; also can be used to cause a physical page to be unmapped (i.e., no memory is assigned to the physical page).
6. Deallocate Pages: Return the pages allocated by an application to the EMM so that other applications may use them. The handle obtained using function 4 is required to deallocate pages.

7. Get Version: Pretty obvious, this call will return an encoded value of 3.2 if the EMM supports version 3.2 EMS.
8. Save Page Map: Saves information about which logical pages are currently mapped so that the current mapping configuration can be restored later. The information is saved in a structure associated with a supplied handle.
9. Restore Page Map: Restore the mapping configuration saved during a previous call to function 8. Again, a handle number is required.
10. Reserved: This is one of those function calls defined in a previous version of the EMS which allowed an application to get hardware specific information. It is now an illegal function call in order to hide the hardware specific information.
11. Reserved: For the same reason as function 10.
12. Get Handle Count: Returns how many separate applications (handles) have allocated expanded memory.
13. Get Handle Pages: Returns how many pages are allocated to a particular application (handle).
14. Get All Handle Pages: Returns a map of how many pages are allocated to each application (handle).
15. This last function call is broken into four separate subfunctions. As a set, the subfunctions provide essentially the same functions as functions 8 and 9 except that a single application can keep (save and restore) multiple sets of mapping configurations instead of just one.

That's it! All the functions anyone would ever need to access expanded memory, right? Well, no. Version 4.0 of the EMS specification more than doubles the number of functions. I'll go in to that later, first I want to describe another memory type.

What is Backfill Memory?

As expanded memory boards were being developed some creative person came up with the idea that if a person needed more memory in their system, they may not already have a full 640K. So they figured out that rather than require said person to buy some sort of memory expansion kit to fill out their 640K and then buy another board in order to get expanded memory, perhaps an expanded memory board could offer both capabilities. They came up with a way whereby an expanded memory board could use some of its memory to fill out the 640K and the rest would be used for expanded memory. The memory used to fill out the 640K address space is referred to as backfill memory. When this capability was first offered, the backfill memory supplied was not expanded memory. In other words, memory supplied by the hardware adapter became conventional memory; it was assigned

memory addresses which could not be changed. That was to change as you will see.

Confusing EMS 3.2 With EMS 4.0

Now if we stopped right here, much of the confusion surrounding expanded memory could be avoided, but no, the 4.0 version of the Lotus/Intel/Microsoft Expanded Memory Specification added a lot of new (and powerful) functionality to the concept of expanded memory. About half of the new features available as described in the 4.0 version of the specification are specific to new hardware features. In other words, an expanded memory board capable of supplying complete LIM EMS 3.2 functionality may not be able to supply some of the added features defined by the LIM EMS 4.0. That, however, does not mean the LIM EMS 4.0 cannot be used with older hardware. This is a very often confused point. The LIM EMS 4.0 specification calls out a number of hardware specific functional enhancements. It also defines a method for the EMM to report which if any of the enhancements are actually available. **Thus, an EMM compliant with LIM EMS 4.0 may work with hardware which does not support any of the additional hardware features defined by that specification.** All it need do is report that only those functions which the original expanded memory boards support are actually available. It would still have to be referred to as supporting LIM EMS 4.0 even though it does not supply the extended hardware features. In fact, it is possible, though somewhat misleading, to indicate that the older hardware is compliant with LIM EMS 4.0 if such an EMM is supplied.

What is "Large Page Frame EMS"?

Having explained backfill memory earlier, the first new feature I would like to cover is something known as large page frame support. You may remember that in version 3.2 there are precisely four physical pages which may be used to map expanded memory. Well, in version 4.0 there may be many more. Any EMM which provides more than the basic four physical pages is said to support large page. Of the 4.0 hardware features, this is the one most likely to be supported by an EMM. Large page frame support is available via many radically different hardware interfaces. Fortunately, the EMM makes it easy for applications to use all those different hardware types without knowing just how different they really are. Ideally, large page frame support means that the hardware is capable of mapping logical page frames to virtually any memory address. The LIM EMS 4.0 document has limited this ideal to the address ranges of 40000 through BFFFF and C0000 through EFFFF. The first of these two ranges appears within the conventional memory address space. Typically,

any EMM which supports large page frame will support those page frames which appear within the conventional address space. Why? That's my lead-in to the next new feature.

What is an OS/E?

Just suppose that there was a program which worked very closely with MS-DOS. And just suppose that program could supply backfill memory with logical pages of expanded memory. Then just suppose that after MS-DOS had fully loaded that memory with applications that this program came along and changed which logical pages were used as backfill memory and informed MS-DOS that magically there was a bunch of free conventional memory again. MS-DOS would then go ahead and allow that memory to be filled with more applications. Then this imaginary program could start swapping the first set of applications with the second set of applications. A third and fourth set could be added. Pretty soon, you have many, many applications loaded in "conventional" memory. Much more than should actually fit. Sound neat? Well it just so happens that the imaginary program does exist. More than one of them in fact. Both Windows and DesqView can play out the role I have just described. In fact this expanded memory feature was designed with programs like them in mind. Programs which manage expanded memory in this fashion are called "Operating System / Environment" (OS/E) programs. Only one such program may be used at a time.

An OS/E has a very difficult burden of responsibility placed upon it. It must be able to swap programs in and out of conventional memory using the EMM even though it has only a limited amount of control over what those programs are doing. One problem which has to be overcome is to insure that the proper interrupt vectors are maintained for each separate program. Another problem is to make sure that the programs are not interrupted during some sort of critical processing like a call to the operating system. One of the more difficult tasks for the OS/E is to insure that Direct Memory Access (DMA) transfers work properly even though memory swapping is occurring. For example, suppose an application set up the DMA hardware to transfer 4K of data from a floppy disk drive to a memory buffer. Then suppose the OS/E decided it was time to switch which application was loaded. Since DMA operations are handled by hardware independent from the rest of the computer, the DMA operation will continue to transfer data even if the buffer is paged out and a new program is paged in. This new program would then be written over with the remainder of the DMA transfer. To prevent such problems, the OS/E has to be very careful in selecting when to switch

which programs are mapped.

What are Alternate Map Register Sets?

A new 4.0 function was designed to make it faster for an OS/E to swap between programs. This new feature is designed with special expanded memory hardware in mind. What the additional hardware provides is the ability to change what logical pages are mapped to each physical page simply by changing one hardware register. In order for this to work there must more than one complete set of hardware mapping registers. One set is used to define what is mapped "right now". Another set is used to define what will be mapped "later". It takes a little bit of work to prepare each of these sets of registers, but it takes almost no effort at all to change which one is active at any particular time. Unlike some of the new features, this feature can actually be emulated using software. In other words, without special hardware the EMM can make it look like the special hardware is present. Of course a software approach is not nearly as fast as a hardware approach, but it does simplify the process of switching a bunch of logical pages back and forth over and over again.

What are DMA Register Sets?

Another new 4.0 function was designed to make it easier to insure that DMA operations did not interfere with the swapping of programs and buffers. This new feature *requires* special expanded memory hardware. The additional hardware provides the ability to define which logical pages are mapped to each physical page whenever a DMA transfer occurs. In order to understand this you need to know a little more about how DMA transfers occur on a PC. DMA hardware is programmed to transfer a block of memory from a hardware device to memory or visa-versa. Once started, a single byte of data periodically will be transferred until the entire operation is complete. The time during which a byte is transferred is called a DMA cycle. DMA cycles occur without any further software interaction. In fact, only the hardware can tell when DMA cycles occur. The special expanded memory hardware, for example, can detect that a DMA cycle is going to occur. It can then change which pages are mapped, let the DMA cycle occur, and then change back again. DMA register sets differ from alternate map register sets only in that they are automatically used during DMA cycles. In fact some expanded memory hardware lets you choose whether or not an alternate mapping register set is used as a DMA register set.

There are two different forms of DMA register set hardware implementations defined. The simplest form provides one mapping register set which is used whenever any DMA cycle occurs. The more complex form allows for multiple mapping register

sets. Each such register set can then be assigned to a specific DMA channel. The original PCs had four such DMA channels. AT class machines have a total of seven DMA channels. By assigning the register set to a specific DMA channel, the automated process of swapping which pages are mapped will only occur during a DMA cycle for the specified channel. Using more than one such register set it is possible to cause all sorts of swapping to occur automatically.

Using DMA register sets, an OS/E can swap which programs are loaded without waiting for some DMA operations to complete. There are still a few problems however. Suppose for example that two programs both need to use the same DMA channel. The OS/E would still have to insure that program swapping occurred only when DMA is idle. This tends to imply that the OS/E must run in a protected operating mode. Perhaps it is problems such as this which have resulted in a very limited appeal to the DMA register set capability. Very few expanded memory hardware solutions include this feature. Even those which include large page frame support frequently do not include this feature. Even fewer EMMs care about supporting DMA register sets. They simply report that they have none and leave it at that.

What is Non-Volatile Expanded Memory?

One of the least used features defined by the 4.0 EMS specification is that of non-volatile EMS handles. If the EMM supports non-volatile handles, it supports the ability to retain data through a reboot process. The best example I can offer of how this feature would be used is that of a non-volatile RAM drive. A RAM drive could be defined such that once you turned on your machine in the morning, the data stored on that drive would remain there until you turned your machine off, no matter how many times you rebooted your machine! The only reasons I can offer as to why this feature is not more readily available are:

Not many EMMs offer this feature. This may be because it is a somewhat tricky feature. Really though, most hardware can be programmed in such a manner as to provide this feature.

The application is responsible for insuring that a "prepare for warm boot" call occurs prior to any reboot of the machine. It is this which is somewhat tricky, if not impossible, to insure. There are usually too many different ways which your computer can be rebooted.

Most programs which would make use of non-volatile memory can use extended memory instead of expanded memory. Providing a "non-volatile" support when using extended memory is feature not offered by any defined extended memory interface. Thus, those applications would have to explain why such a feature

is only available using expanded memory.

What is a "Raw Page"?

For certain, the least used 4.0 EMS feature is the ability to have the size of expanded memory pages be something other than 16K. For example, an expanded memory hardware device could have the size of an expanded memory page be 4K. In that case, a 4K page would be called a raw page and a 16K page would be called a standard page. In order to be compatible with the 3.2 specification, the EMM would have to normally operate in groups of four raw pages at a time in order to make standard pages be 16K each. Then, if an application chooses to use a few of the extended functions, the EMM could report that in fact the hardware page size is not 16K but 4K. It could then operate on 4K raw pages instead of 16K standard pages. However, since it is so absurdly difficult for an EMM to keep track of all this, even if there were hardware which provides pages in 4K sizes, the EMM will never tell the application that fact and instead will simply report that raw pages are the same size as standard pages (i.e., 16K).

LIM 4.0 Functions

In order to completely understand the functions added by EMS 4.0 one must read the full specification. However, a general idea can be grasped by a simple description of the major functions provided. EMS 4.0 supports all of the 3.2 functions in addition to the ones described below. Since the 3.2 functions are numbered 1 through 15, the 4.0 additions start with function 16:

16. Function 15 provides a means of saving and changing the mapping of all hardware pages simultaneously. Function 16 allows a subset of all hardware pages to be modified. Other than that it functions exactly the same.
17. Similar to function 16, this function provides the ability to map more than a single page in a single call. It differs from function 16 in that the logical pages mapped are explicitly specified rather than restoring what was mapped at an earlier point in time.
18. With this function it is possible to change the number of logical pages allocated to an application without relinquishing the handle used to claim memory for the application in the first place. This is useful if more memory is required and the memory already allocated contains information which must not be discarded. Conversely, if an application is suspended it may wish to release unused pages until it is reactivated.
19. Function 19 is used to define whether the memory allocated to a handle should be non-volatile or not. The application can also use this function to find out if the EMM even supports non-

volatile memory or not.

20. If an application is going to use the non-volatile memory feature, it pretty much has to assign a name to the applications expanded memory handle. This name can then be used to determine if a handle still exists after the system is rebooted. In other words, handle names are used to distinguish warm boots from cold boots.
21. This function is used to locate a handle by name. Again it is used by those applications which use non-volatile memory. It is also used by programs such as the MS-DOS 4.0 MEM program which can display the name assigned to each handle currently allocated by the EMM.
22. This function provides the ability to change what pages are mapped and then jump to any location in memory. Presumably this function would only be used if the code being jumped to resides in expanded memory which is mapped prior to jumping to it. Furthermore, such a function as this would not be necessary unless the code performing the call resides in expanded memory which is unmapped prior to the jump being performed. One ideal use of this function is for an overlay manager which allows overlay code to be located in expanded memory.
23. Of course, if a special jump operation is provided, a call operation also makes sense. This function provides just such a capability. However, it provides it with sort of an interesting twist. Using this call it is possible to define what pages are mapped both prior to and after a "call" operation is performed. Normally, one would expect that after a call completes the mapping would return to what it was prior to the call. That is possible using this function, but it is also possible to have a completely new set of pages mapped upon "return" from the call. Note, however, that the return address is the address of the instruction following the call to the EMM.
24. This function is the most intricate function implemented by an EMM. It is a general purpose memory move/exchange function. Using this function it is possible to move or exchange regions of any combination of conventional and expanded memory. The amount of memory moved or exchanged can be anywhere from one byte up to one megabyte. Overlapping moves are handled so as to insure that the destination is an exact copy of the source even though the source is modified in the process. One thing which makes this function very useful is the fact that it is possible to address logical pages which are not currently mapped. The driver will map pages as

required and restore the mapping prior to returning to the application.

25. This function is used to determine if the EMM supports large page frame. Both the number and address of all physical pages accessible by the driver are returned by this function. Every physical page has both a number and address associated with it. Both can be obtained with this function.
26. This function returns information about what hardware specific features are provided by the EMM. For example, the number of alternate map register sets and/or DMA register sets are returned by this procedure.
27. This function must be used if an application has decided to use raw pages instead of standard pages. Since no EMM is likely to actually make a distinction between the two, it is unlikely that this function is ever really used to allocate RAW pages. There is one unique capability with this function however. Using this function it is possible to allocate an EMS handle with zero pages of memory. That is not possible using function 4.
28. Use of either alternate map register sets or DMA register sets is requires this function. Since normal applications do not use these features I will not describe the specifics, especially since there are nine separate subfunctions.
29. This is the function which an application must invoke prior to allowing the system to reboot when that application is using a non-volatile memory. Note that the EMM itself does not guarantee anything other than the fact that if this call is made at the appropriate point in time, the EMS configuration will be maintained throughout the boot process, but only for those applications which have non-volatile handles assigned.
30. This function is used by an OS/E to restrict access to functions 26 and 28. It is also used to insure that only one program can become an OS/E at a time.

What is the Difference Between Expanded Memory and Extended Memory?

I don't want to describe all of the details associated with extended memory, but because the name is too similar with expanded memory, confusion is prevalent and so some distinction is appropriate. Extended memory is only possible on 80286 or greater. That is because extended memory starts at a memory address not available to 8088, 8086, and 80186 based machines. All memory which is mapped at or above address 00010000 is called extended memory. Extended memory is not paged. It cannot be unmapped or remapped to appear at a different hardware address.

This leads me to a simple rule which can be used to remember which is which. Expanded memory is paged. Note the 'p' in both expanded and paged. Extended memory is not paged. Note that the word extended has no 'p' in it. Since paging is really what distinguishes expanded memory from any other form of memory, this is a good rule to remember.

What are the Differences Between Hardware EMMs and Software EMMs?

So far I have not even mentioned anything about software EMS. I did state that I would clear up some confusion in that area so let me begin to do so. The first EMMs used hardware memory managers to provide expanded memory. With the introduction of the 80386 there is now another way. The memory controllers embedded in the 80386 SX, 80386 DX, and 80486 all have a mode which enables a virtual address space to be defined which differs from the actual hardware address space. A virtual address used by a program

is converted to a hardware address without the program being aware of the conversion. This virtualizing of address spaces enables operating systems environments with multiple virtual machines. For example, multiple machines, each with 640K of conventional memory, can be simulated.

The virtual memory mapping in 386 machines relates to expanded memory because drivers can be written to use this feature to simulate expanded memory hardware. To expanded memory applications, there is virtually no way to distinguish such a device driver from one which uses actual expanded memory hardware. An EMM which uses the 386 memory manager is said to be a software EMM rather than a hardware EMM. Another commonly used name for to software EMMs is "LIMulator".

LIMulators have the advantage of working on all 386 or greater machines with extended memory. Extended memory is used by the LIMulator as if it were expanded memory. By that I mean, a LIMulator will use the 386 memory man-

ager in virtual mode to map extended memory to an EMS physical page. Note that the memory is only mapped in the virtual address space of the 386. The hardware addressing is unchanged. Because a LIMulator must run in virtual mode, and because it must setup special handlers, LIMulators tend to slow down your system by some significant percentage.

Until Next Time . . .

Well, I think I have fully answered all of the questions I have set out to answer about EMS. Now if I could only find the time and energy to start an article on upper memory. Then there's the details associated with extended memory such as HMA and EMB's. Of course, there's also slush memory or shadow RAM. Then to really bring it all home there's all of the specific differences between all of the ZDS computers and how all of these different memory management schemes fit together (if at all). Unfortunately, finding time and energy is not my strong suit. ✱

Continued from Page 18

drive will trash that drive for good. I've found such rumor without foundation, but admit truthfully that I did not try all four different drives yet.

DETECT, PREP, PART, ASGNPART, FORMAT as with any other hard drive. Every so often the "too many errors" message will come up. I've found it can be ignored without any ill effects.

The last thing to do is to transfer the system to the hard disk using the SYS command. If you added an outside floppy drive, run DSKSETUP also. Now, before you pat yourself on the shoulder, shut off your eaZy PC, remove any disk from drive "A" and turn it on again. It will start with a system test, then make 2 attempts to boot from drive "A" before it boots from the hard drive.

What if it does not boot? One possible cause can be a bad spot at the outer cylinders of the hard drive. Well, the beginning of the first partition does NOT have to be the 0 cylinder. If you suspect a bad spot there, try 10 as the beginning of the first partition. Usually, it will cure the problem.

Now pat yourself on the shoulder, for you got it made!

Suppliers of Stuff for the eaZy PC

FBE Research Co., Inc.
Dave Brockman
P.O. Box 68234
Seattle, WA 98168
(206) 246-9815

Quikdata, Inc.
Henry Fale
2618 Penn Circle
Sheboygan, WI 53081-4250
(414) 452-4172

Used or Surplus Stuff

Surplus Trading Corp.
Les Turk
2700 N. M-63 Hwy.
Benton Harbor, MI 49022
(616) 849-2995

Other Electronic Stuff

Digi-Key Corp.
701 Brooks Avenue South
P.O. Box 677
Thief River Falls, MN 56701-0677

Jameco Electronic Components
1355 Shoreway Road
Belmont, CA 94002

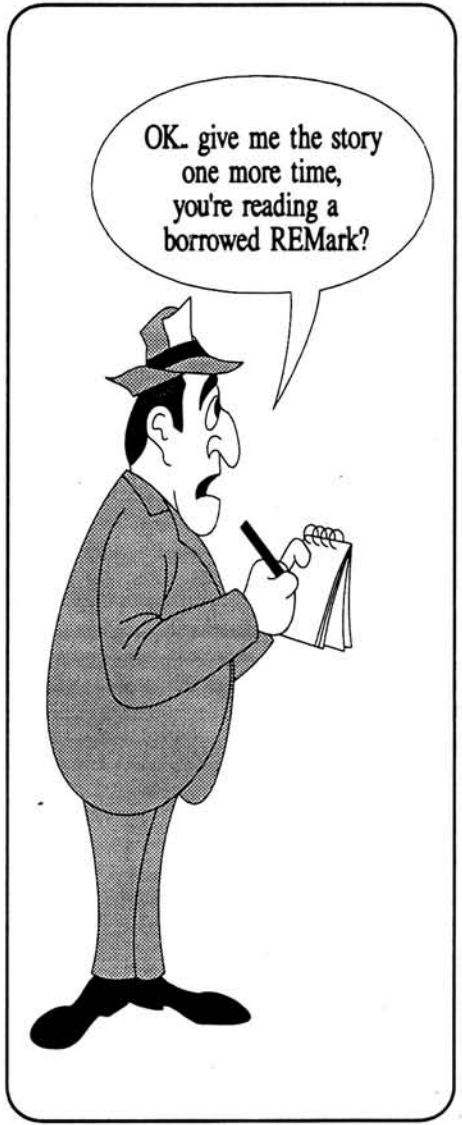
JDR Microdevices
(Unlisted address)
(800) 538-5000 ✱



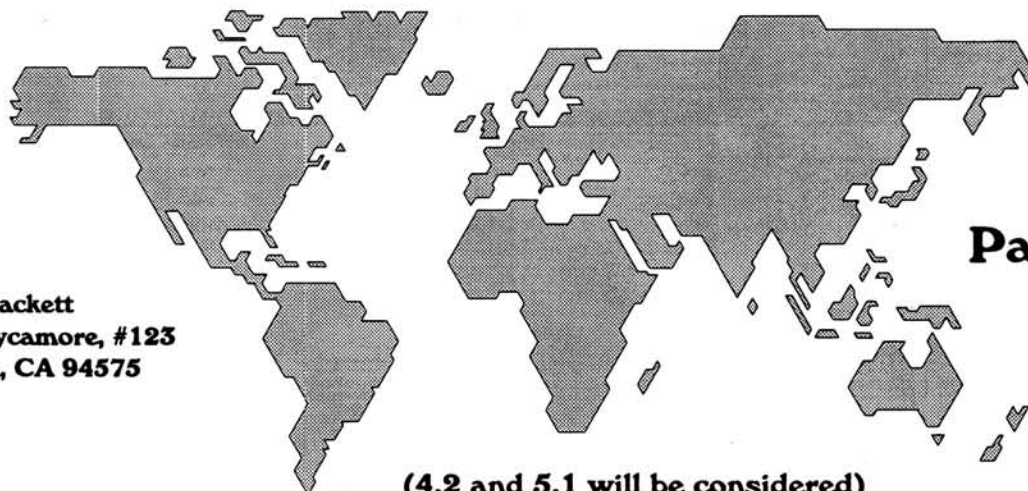
MOVING?
Don't miss a single issue of REMark!

Please let us know 3-4 weeks
before you move. Call
(616) 982-3463 or write:

Zenith Data Systems
P.O. Box 217
Benton Harbor, MI 48023-0217



The World of WP50 and Its Wonders



Part VI

Salli Brackett
2201 Sycamore, #123
Antioch, CA 94575

(4.2 and 5.1 will be considered)

Footers are lines of text before the bottom margin that print on every page. Headers are lines of text after the top margin that print on every page. You cannot see the headers/footers text on the screen. To see them, you need to use view (Shift-F7, v(iew)). Footers/headers can be many lines long. Footers by default print one line after the text. If you want more, you must put a blank line in the footer before your text. Headers by default print one line before the text. If you want more, you must put a blank line after the text in the header. To manipulate footers/headers it is best to put them all at the beginning of your document. If the first page doesn't need headers and footers, you can use suppress page (explained later).

The subject in this article is headers/footers. This feature has many applications other than the obvious, page numbering. We will explore several applications and options in placing footer/headers where you want them on the page. If you have any applications other than the ones mentioned, please write me. I would love to hear from you.

Placement on Page

To have the footer/header appear on the same page, the code (Reveal codes) must be before any text. Remember, the default top/bottom margin is 1". That means your footer will appear one inch from the bottom of the page, your header will appear 1" from the top. If you wish the footer closer to the bottom of the page or the header closer to the top, follow the explanation below.

1. For a footer, make the bottom margin less, i.e., .5". Remember to place your footer code after the margin code.
 - Press Shift-F8, p(age), m(argin)
 - Press ENTER
 - Type .5
 - Press ENTER
 - Press F7

Then, with reveal codes on, cursor to the right of the margin code. Create your footer (See *To Create a Header/Footer*).

[T/B Mar: .5", .5"] [Footer A: Every Page. . . .]

The logic behind this is simple;

WordPerfect reads just like we do — left to right. Therefore, it will read your margin change first and then the footer.

The disadvantage to this is that your text will also be further down on the page. To avoid this, place a couple blank lines before your text in your footer.

This is the end of the text.

wp506.art

2. For a header, change the top margin. Remember to place your header code after the margin code.
 - Press Shift-F8, p(age), m(argin)
 - Type .5
 - Press ENTER twice
 - Press F7

With reveal codes on, cursor to the right of the margin code. Create your header (described on Page 7).

The text will start higher on the page. Just add two or three blank lines after the text in the header. This way the text will start lower on the page.

WordPerfect — Footers ... Page ^B

This is the beginning of text.

WordPerfect will always save enough

room for your header/footer and your margins, no matter how many lines. Therefore, the more lines in your header/footer, the fewer you have for text.

I find using the inch designation for units of measure complicates placing footers/headers; therefore, I use 4.2 units of measure. These measurements are in spaces and lines. Now you can change footers/headers by lines instead of fractions of an inch.

- To change the measurement:
- Press Shift-F1, u(nits)
 - (WP5.1 - e(nvironment, u(nits))
 - Type d(isplay), u, s(tatus), u
 - Press F7

1 inch converts to:

	COURIER	HELVETICA 12 POINT
LINES	6	5.15
SPACES	10	11.11

Therefore, on 8-1/2" x 11" paper:

	COURIER	HELVETICA 12 POINT
LINES	66	57
SPACES	85	94

The left/right margins read:

	COURIER	HELVETICA 12 POINT
LEFT	10	11.11
RIGHT	75	84

The top/bottom margins read:

	COURIER	HELVETICA 12 POINT
TOP	6	5.15
BOTTOM	6	5.15

You only see the text on the screen; therefore, the Ln number on the status line (lower right-hand corner) refers to the number of lines of text, not the placement on the page. You would have to add your top margin and the number of lines in your header and the extra line after the header to know exactly where your text will start. For example, say you have a 1" margin (6 lines) and a header with 2 lines. Add the line between the header and the text. This means your text would start on line 10 or 1-1/2" from the top of the page.

If you are trying to get text all on one page and there are two lines going to the next page, decrease your bottom margin or make sure the header or footer has no blank lines.

Applications for Footers

Filename in small print

wp506.art

I recommend using this in all files. Even in business, it has become acceptable to put file names on letters and memos. With a filename on all documents, you can then use a numbering system for files and retrieve them easily. For example, if you create files for several people, use their initials and a number, i.e., cl.101. I recommend large documents or forms that will be frequently edited, be given a text name, i.e., contact.frm. Printing the filename in small print makes it less obtrusive.

Once in footer edit menu:

- Press Ctrl-F8, s(ize), s(mall)
- Type in name
- Press F7 twice

Filename, page number, and date

wp506.art ^B 07/07/91

This is appropriate for multipage documents. The date is optional for documents that will be edited frequently. The line is optional, but it does give a nice separate affect to the footer.

There is a blank line before the text. The line is created with Graphics—horizontal line, the filename is in small print, page numbering is centered, and the date is right

aligned, using date code.

When you use the date code, the date will be updated by using the date from the system (DOS). The default format for the date is (January 1, 1991). This must be changed to 01/01/91.

Once in footer edit menu:

- Press ENTER
- Press ALT-F9, l(ine), h(orizontal)
- Press F7 (The default settings give you a line across the page at the cursor position)
- Press ENTER
- Press Ctrl-F8, s(ize), s(mall)
- Type in name
- Press Shift F6
- Press Ctrl-B (page numbering)
- Press Alt-F6

To change the date format:

- Press Shift-F5, f(ormat)
- Type %2/%1/5

To see how to read these codes, study the screen. All the information needed for the different formats is in this menu.

- Press ENTER, c(ode)
- Press F7 twice

© Copyright 1991

This might be needed on the cover of a multipage document. You could use footer B for this, then discontinue it on the next page.

- Cursor to the next page
- Press Shift-F8, p(age), f(ooter), b, d(iscontinue)

Using a footer is preferable to spacing down and guessing where the placement is on the page. Using a series of carriage returns to obtain this at the bottom of the page will cause problems later on. If you add to the text above the carriage return, the line you wanted at the bottom will now be on the next page. I encourage you always to think ahead when word processing.

The following are applications for headers.

Page headings for memos and letters

John Smith
Page ^B

This header, like all headers, goes at the top of the document. Since you do not want a header on page one, use suppress (see page X). You can create a format file for memos and letters, putting in this header with suppress page. The advantage to this is that you don't have to create a header every time you need it. If there isn't a second page, the header doesn't disturb the rest of the document.

To put two extra lines between the header and text, press ENTER twice after the text.

Subject headings for different chapters, as in a manual.

WordPerfect — Footers Page ^B

WordPerfect — Headers Page ^B

Create the first chapter header at the top of your document.

With reveal codes on (F11), place the cursor on the header code ({header : a : every page}).

- Press DEL, r(estore)
 - Cursor to beginning of next chapter
 - Press F1, r(estore)
- Continue to each chapter and use the restore code (F1, r(estore)).

Now you have an exact duplicate header at the beginning of each chapter. Next, you need to edit each code.

Cursor to the right of the header code in the second chapter.

- Press Shift-F8, p(age), h(ader), a, e(dit)
- Change the text
- Press F7 twice

Edit each header until all are done.

This only has to be done once. As you edit the document, the header stays with the chapter and the page numbers change accordingly.

This method can also be used to have headers that refer to subject headings instead of chapters. Place the header codes next to the subject headings, no matter where they are on the page. Edit each subject header in the same manner as above.

Remember: If there is a header/footer placed after the text on a page, WordPerfect will use it on the next page.

You can create any of these footer/headers you would use on a regular basis in a style. Then create a macro for editing footers and one for headers. This way you have easy access to your footers or headers and any changes needed.

To Create a Header/Footer

Remember to be at the top of your document.

- Press Shift-F8, p(age), f(ooter), a

There are 3 ways to create:

- Every page
- Odd page
- Even page

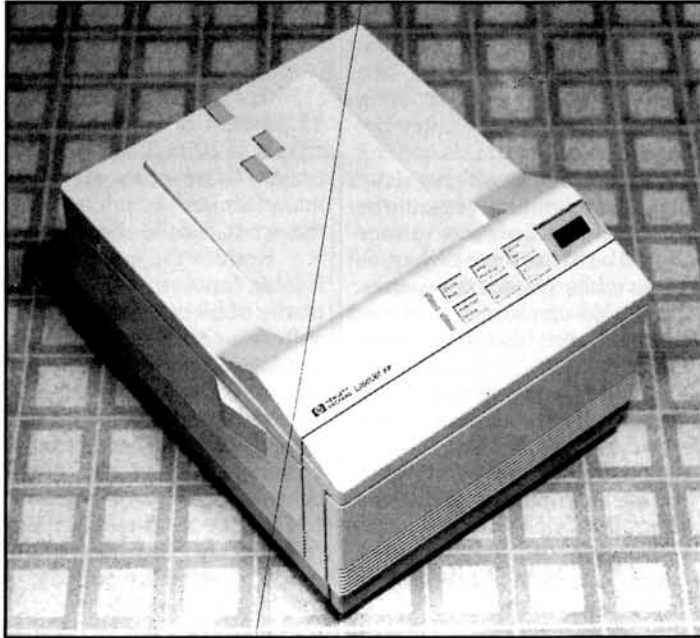
Every page is used if the document is printed on single sides of the paper.

Odd/even page is used if the document is printed on both sides of the paper as in a book. If you use this method, footer A could be Odd page and footer B could be Even page.

- Press p(every page)
- You now have an edit screen. You can type anything you wish for the footer.
- Press F7 twice

If you change the default font for your document, place it in the document format, not in the document itself. The footers and headers work off the initial document font (Shift-F8, p(age), d(ocument), f(ont)). **Continued on Page 37**

The Hewlett-Packard LaserJet IIP



*William Wills
202 Meadowbrook Avenue
Boardman, OH 44512-3004*

Are you thinking about replacing that slow, noisy dot matrix printer with the speed and quality of a quiet laser printer? Despite the fact that it only prints four pages per minute, the \$1295 (street price around \$950) Hewlett-Packard LaserJet IIP is not a laser printer to cross off your list. Like the HP LaserJet Series II printer, the IIP will give you the same high quality, reliable laser printing from any IBM compatible PC and it will do a few things the Series II and other laser printers can't.

The IIP's print quality is just as good as the Series II, and it comes with more internal fonts, adding 12-point Courier Italic and a set of 10-point Courier fonts. Like the LaserJet IID, it can also rotate any portrait font to landscape orientation. With its small footprint (14" wide by 16" deep by 8" high) and a weight of 22 lbs., the IIP is truly a personal laser printer. And since its no bigger than a dot matrix printer, the IIP will allow you to place it on any corner of your desk.

HP has gone far to ensure that the setup of the IIP is not difficult. The Canon LBP-LX engine combines the drum and toner in one mess-free cartridge. You open the printer's front door, pull a plastic tab from the cartridge and slide it into the printer, and close the door. If you do run into difficulty, you'll find an illustrated *Getting Started Guide*, which will lead you by the hand through the setup.

The front mounted control panel, which consists of a two-line liquid crystal display (LCD) panel and six large, well

marked buttons, lets you easily specify the paper size and the print mode. The IIP makes you select fonts by number instead of by name, but it prints a numbered font list at the press of a couple of buttons. In addition to its well organized, easy to understand manual, you'll find a second manual that tells you how to use the IIP with the most popular software.

The IIP warm-up time is less than 60 seconds from a cold turn on. It prints the first sheet in less than 40 seconds after it receives the data from the computer and the subsequent sheets print as many as four sheets per minute. It will take longer to print the first sheet if you are using downloadable fonts or you are printing graphics, but this is true with any laser printer.

If you're going to use the IIP for graphics or plan on using it for Desktop Publishing, the IIP can create clean, bold graphics at 300 dpi. Though with its 512 Kbytes standard memory, you can only produce full page graphics at 150 dpi or half sheet graphics at 300 dpi. If you plan to mix high resolution graphics and downloadable fonts, you'll definitely need to buy an add-on memory board.

The IIP comes with 512 KB of standard memory (365 KB of user accessible memory at power up), but you can easily install a 1 MB or a 2 MB memory board in one or both of the internal memory slots. The two memory slots allow you to install up to 4 MB of memory. When you're ready to upgrade your memory, you might want to shop around first. HP charges \$495 for a 1

MB board, \$990 for a 2 MB board and \$1980 for a 4 MB board. The street price for third party memory upgrade is around \$85 for a 1 MB memory, \$135 for a 2 MB memory board and \$245 for a 4 MB memory board. Before you run out and upgrade the memory in your IIP, consider how much memory you will need at a later date. If you will need 4 MB a few years down the road and you purchase a 1 MB memory board, you will not find a 3 MB memory board.

The lower price tag does bring with it some drawbacks — it prints only 4 ppm compared with the 6 ppm or 8 ppm of most other laser printers in its price range. In its default configuration, its only paper feed is a flip down fifty sheet maximum multi-purpose tray. In the down position, the tray adds about 9 inches of depth to the printer. It also has only one cartridge slot.

The IIP does not come with a paper counter either, but by following these steps you can get a paper count: Holding down the ON LINE, MENU and ENTER keys at the same time while turning on the printer and allowing time for it to warm up. You will be presented with a blank display in the window, and both indicator lights will be illuminated. Now press the FORM FEED key and the ENTER key. The display will show "00 SMODE" and underneath will be the page count. Then press the ON LINE key to put the printer in service.

With the addition of Hewlett-Packard's \$295 (street price around \$150) optional lower paper cassette, which comes with a

Courier 10 point
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 abcdefghijklmnopqrstuvwxyz
Courier 10 point Bold
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 abcdefghijklmnopqrstuvwxyz
Courier 10 point Italic
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 abcdefghijklmnopqrstuvwxyz
Courier 12 point
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 abcdefghijklmnopqrstuvwxyz
Courier 12 point Bold
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 abcdefghijklmnopqrstuvwxyz
Courier 12 point Italic
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 abcdefghijklmnopqrstuvwxyz
 Line Printer 8.5 point
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 abcdefghijklmnopqrstuvwxyz

Hewlett-Packard LaserJet IIP Internal Fonts

letter tray, the IIP's paper handling capability is increased to 300 sheets. The cassette cabinet screws onto the bottom of the printer, adding less than 2 inches to its height and holds 250 sheets. The lower cassette also comes with an illustrated *Getting Started Guide*. Besides the greater capacity, the cassette allows the IIP to do something the Series II and other laser printers can't, and that's print from two different paper bins. You can use letterhead in the Multi-purpose tray and plain paper in the lower cassette tray. Other trays, including legal-size, A4 size and executive size are also available at prices ranging from \$69 to \$89. The street prices are 10 to 15 percent lower.

In its normal configuration, it can only handle a maximum of five envelopes from the multi-purpose tray. With the optional envelope tray, it will accept up to 20 envelopes. It can also print up the four different sizes of envelopes using either the multi-purpose tray or the lower cassette tray. With the envelope tray, you can print a letter from the MP tray and automatically address and print an envelope from the lower cassette.

The IIP outputs pages face down (correct order) atop the printer via an S-shaped path, or face up (reverse order) through a slot at the front of the printer via a C-shaped path. You can print envelopes using the face up feed with no crinkling or excessive curling. Though it doesn't have a straight through path for heavy stock, the IIP can handle up to 28-pound paper, labels and overhead transparencies through the front slot, according to Hewlett-Packard.

Several types of media can be used in the IIP, provided certain guidelines are

met. Use a high quality conventional white xerographic paper (also called photocopy or xerocopy paper) for most printing applications. This type of paper will ensure good image transfer and tone fixing without excessive curling.

You can also use color paper in the IIP, but don't use paper with a colored coating that has been added after the paper is produced. This coating will not be able to withstand the fusing temperature without deterioration. Colored paper should be of the same high quality as white photocopy paper. Most manufacturers of xerographic paper also make color paper.

Letterhead or Bond papers generally have a water mark and often use cotton fiber. These papers tend to have a rougher surface than photocopy paper. The cotton fiber may cause smearing, because the toner will

not fuse to the paper. These fibers can also cause premature wear to the drum. However, your IIP will print on many of these satisfactorily, providing they are a quality grade of paper. Also, some manufacturers are now making laser grade bond papers.

The IIP will also print labels and envelopes. Use only labels recommended for laser printers. The adhesives used on laser labels are an acrylic-based emulsion, which is more stable than the adhesives used with labels for a dot matrix printer and it will not give off hazardous emissions when heated. No adhesive should come into direct contact with any part of the printer because the label stock may stick to the photosensitive drum or rollers.

When printing envelopes, use only high quality envelopes with thin, sharply creased, double-sided seams. The adhesives on the envelopes should not scorch, melt or release hazardous emissions when heated. Always use the face up output tray when printing labels or envelopes to avoid excessive curling and jamming.

The \$95 (street price around \$67) EP-L toner/photoreceptive drum cartridge makes maintenance a breeze, too. Every 3500 pages (at five percent coverage), you simply pull out the old toner cartridge and slide in a new one. Run the self-test by holding down the ALT key and the TEST key. The self-test will print two pages, the first sheet is the self-test report and the second sheet is cleaning paper. You place the cleaning paper in the Multi-purpose tray and run another self-test. It takes a whole two minutes to complete. If you do have the optional memory installed, more time is required for the self-test.

If you are worrying that a one piece

drum and toner cartridge is going to cost you more to run than the separate drum and toner cartridge of other laser printers . . . don't. The IIP is less expensive. If all printers work as the manufacturers claim they do, for every 3500 pages a IIP cartridge would cost you \$95 or 2.71 cents per page, less the paper. For the separate drum and toner type printers, printing 3500 pages uses approximately 2.33 toner cartridges (1500 pages per cartridge) at \$29 each and 35 percent of a drum at \$149 each. The cost is \$120 or 3.5 cent per page, less the paper. These costs are all based on the manufacturer's list price. Using street prices, the cost would be considerably less.

Besides the internal Courier and Line Printer fonts, the IIP can accommodate plenty of bit-mapped fonts via cartridge or software. All the LaserJet Series II font products, marketed by Hewlett-Packard and others, will work with the IIP. Hewlett-Packard's cartridges begin at \$99 and go up to \$300 for the 65-font ProCollection cartridge. Prices for Hewlett-Packard downloadable fonts range from \$95 to \$225. Check the street prices and the cartridges and downloadable fonts of third party manufacturers; you will find that they're considerably less.

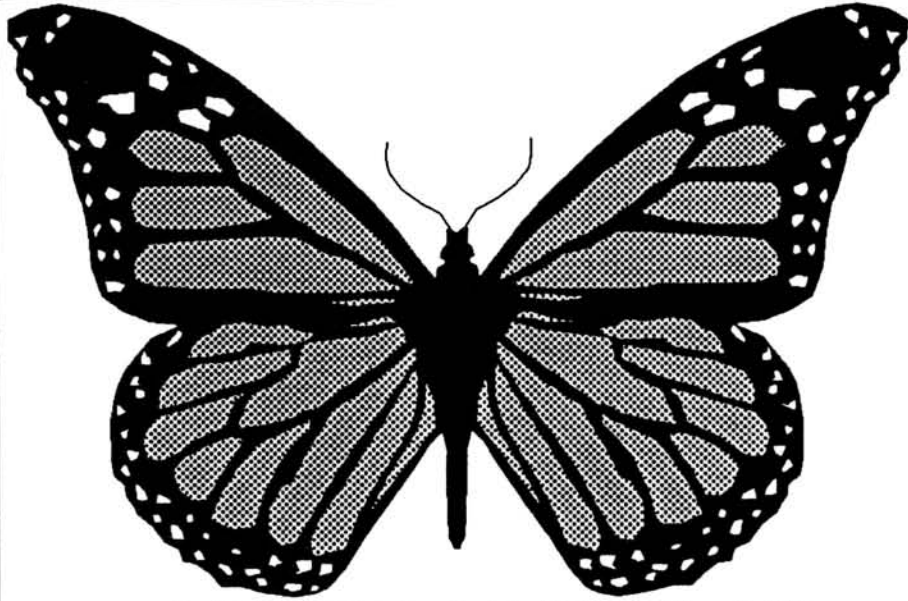
At 4 ppm, the Hewlett-Packard LaserJet IIP is not the printer you should choose if speed is of the essence. If you want to add a print quality enhancement board, such as the DP-Tek LaserPort GrayScale or LaserMaker LX6, you should look elsewhere because the IIP does not support the additional I/O port required for these products.

With an optional cartridge, you can have the IIP emulate an Epson FX or an IBM Proprinter dot matrix printer. You can even emulate Adobe PostScript with a \$695 PostScript cartridge and at least 1 MB of memory. The only problem is that you cannot use a font cartridge with the PostScript cartridge, since the IIP has only one cartridge slot. Again, check the street prices and those of third party manufacturers.

Hewlett-Packard warrants the LaserJet IIP for one year, this also includes the label, from the time you receive it. It also provides a selection of sensibly priced service contacts for the IIP. Besides the warranties Hewlett-Packard also provides you with Personal Peripherals Assist Line for immediate help.

If you would like more information about the Hewlett-Packard LaserJet IIP or about products for it, you can write or call: Hewlett-Packard
 19310 Prumeridge Avenue
 Cupertino, CA 95014
 (800) 752-0900.

Memory Upgrade
 Simple Technology, Inc.
 E. 17th Street, Suite 202



Sample of Clip Art Printed on the Hewlett-Packard LaserJet IIP

Continued from Page 34

To Suppress Page

This command literally means to suppress headers, footers, or page numbering on *this page only*. If you have a multipage document with a cover page and table of contents, you don't want page numbers on these pages. The footer is at the beginning of your document (the cover page). You need to suppress the headers and/or footers on the cover page and the table of contents page.

- Cursor to top of page you wish to suppress
 - Press Shift-F8, p(age), s(upp)ress
 - If you want to suppress all footers and headers, press A(II).
 - Or you can suppress just a footer or header (read the menu).
 - Press F7 to exit
- If you need to suppress on another page, use *down and dirty copy*.
- Cursor to the suppress code
 - Press DEL y(es)
 - Press F1, r(estore)
 - Cursor to the top of the page you wish
 - Press F1, r(estore) *

Santa Ana, CA 92701
(800) 367-7330 Ext. 55

First Source International, Inc.
36 Arognaut, Suite 140, Aliso
Viejo, CA 92652
(714) 588-9866

Downloadable Fonts
Bitstream
215 First Street
Cambridge, MA 02142
(800) 237-3335

AFGA Compugraphic
90 Industrial Way
Wilmington, MA 01887
(800) 873-FONT

Artech Software
5964 La Place Court, Suite 125
San Antonio, TX 78216
(800) 346-9873
(512) 490-2240

Font Cartridges
Elite Business Applications, Inc.
28 Route 3 North
Millersville, MD 21108
(301) 987-9050

Datascan, Inc.
One Middle Street
Portsmouth, NH 03801
(603) 436-2226

IQ Engineering
685 North Pastoria Avenue
Sunnyvale, CA 94086
(800) 765-FONT

Pacific Data Products
9125 Rehco Road
San Diego, CA 92121

(619) 597-4626

Printer, Toner, Lower Cassette and Trays
Printer Plus
14524-I Lee Road
Chantilly, VA 22021
(800) 877-4683

The Printer Connection
8394 Terminal Road
Newington, VA 22122
(800) 622-7060 *



"GUESS WHAT? OUR TALKING COMPUTER CALLED ME A &@*#!"

Getting the Most From Your Computer

Part 5

John Lewis
6 Sexton Cove Road
Key Largo, FL 33037



In part four of this series, I presented the code for a calendar utility that has been found to be quite useful in the past. Every article is designed to present the code to a truly useful program within each installment, while providing a vehicle for the discussion of various aspects of programming in Pascal.

I am going to deviate a little for this article and attempt to deal with two facets of programming. The Turbo Pascal Debugger will be added to our agenda. Debugging is a fact of life for any programmer that gets involved in the creation of unique software and is an integral part of program development. Another very important detail that is often overlooked is the fact that a session with the debugger can do much to enhance your knowledge of how a program works by "getting inside" the source code while it is executing. Of course, this is an impossibility since the source code does not execute, but the Turbo Debugger simulates this very operation by making it appear that it (the source code) is, in fact, executing. A very powerful tool, one that is sure to promote your understanding of programming in Pascal. As a case in point: Last fall I bought Turbo C++, not because I needed the newer compiler, but because it supported the debugger and my old "C" compiler (Turbo C, version 1.5) did not. As you have doubtlessly surmised, I work with both languages, "C" and Pascal.

I have taken the liberty of yet another deviation from the format employed in past articles. The program listing presented here is not intended for inclusion within the ongoing project program, rather it is a "stand alone" version for reasons which will become obvious as we get into the uses of this month's presentation. We will get back to adding modules in the next issue.

This month's program required several sessions with the debugger before it would behave in the manner desired. The development time was substantially reduced through its use (not to mention the frustration level). "SortDir", although recently created, has already proved to be of great value to me. I hope that you agree with my evaluation. Its usefulness will be in direct proportion to the number of directories/files you have on your disks. Those of you who have recently acquired your computers and have only a few files on your disks may yawn a bit with the prospect of inheriting a utility which displays a sorted directory; however, that situation will change as soon as you add some additional software to your inventory.

There are a number of utilities available that perform the same or similar functions to this month's listing: "SortDir". Most of them have one rather large drawback; they sort the directory and write the new version to disk. What's wrong with that? Nothing really, but it requires that the operation be repeated each time the disk is updated. That might seem like a small price to pay, but if you have a lot of disk activity involving moving/creating files, then you will be wanting a utility that presents your updated directory with each call. I discovered a utility, included with my copy of Turbo Pascal, that almost filled my needs. It is called "DirDemo.pas" and is included with the example source code on my disks from Borland (Turbo Pascal, Version 5.5). I performed a little surgery on Borland's version and had it doing what I wanted, almost.

I prefer to access a program without entering any command line parameters. "DirDemo" required that you specify the field to sort on "-N, -S, -T" (name, size and time, respectively), plus the unique parts of the filename and extension, defaulting to

"*.*" (the "*" symbol represents a "wild card") if none are given. A typical command line might look like this: "DirDemo - N *.Pas". The result would be a listing of the default directory in alphabetical order (by file name) showing all the files with a "Pas" extension. The surgery involved eliminating the first command line parameter, defaulting to sorting by name and allowing the use of specific names and/or extensions if desired. I was delighted with the new configuration. I had a program which did almost exactly what I wanted it to do. I added a feature that allowed the display of 23 lines at a time (a screen full) and liked the result even better. Only one problem remained. I like to view a directory that displays all the sub-directories first, then the files. This makes it much easier to find a given file/directory. You are probably thinking: "what does this guy want — for free yet?"

Well, to be perfectly honest, what I wanted is what you are getting in this month's program. It displays a sorted directory, in alphabetical order (by name) with the directories above and the files below. Actually, there are two listings which are concatenated, since "\Turbo" (a directory) would be above "Remark.txt" (a text file). You may be thinking that this program is a bit of "overkill", but my \TP directory has 240 files in it, not counting the files in "sub-directories". Finding a particular file in an unsorted directory can be rather frustrating and time consuming.

If you are thinking that it is time that I rearranged my directories a bit, you would be right; however, with my latest software tool, the chore is much less urgent.

"SortDir.Pas" is an entirely unique program, incorporating some features found in Borland's "DirDemo", but using a different methodology. I used a different method of dynamic memory allocation for "SortDir"

since this listing will serve as an introduction to "linked lists", a programming technique enabling access to memory locations in a sequential manner.

We have neglected the discussion of memory allocation for variables until now, but this month we will focus briefly on static variables and then turn our attention to allocating memory on the fly, using only what is required by the program data.

Before we get into discussing the actual code and its logic, let's embark on a short trip through some of the features of DOS. As I have stated before, Bill Adney, who writes for this magazine, is the resident expert on DOS (MS-DOS) and his articles provide an excellent reference library on that subject. Since you may not have access to many past issues of "REMark", I will outline some of the commands as they apply to the use of "Sortdir".

Immediately after "boot up", assuming there are no directions to the contrary in your AUTOEXEC.BAT file, your computer will display the default drive and, depending on the contents of your AUTOEXEC.BAT file, the default directory (usually the "root"). This could take the form: "A:" or, if you are using a hard drive, "C:". If your AUTOEXEC.BAT file contains the instruction: "PROMPT \$P\$G", your screen will display A:\> or C:\> instead of the example alluded to above. In the pursuit of simplicity, we will confine this discussion to drive "C". Those of you who are using floppy drives will simply need to substitute the appropriate drive letter in the examples to follow.

Before we continue, let me suggest the creation and inclusion of an AUTOEXEC.BAT file, detailed above, in your root directory if one does not already exist. Your computer always looks for an AUTOEXEC.BAT file in the root directory upon boot-up. Including some pertinent instructions there can make subsequent operations much easier. If you lack an editor, you can use your Turbo Pascal editor to make the above change. Assuming Turbo resides in the \TP directory and you are currently in the root directory, use: \tp\turbo AUTOEXEC.BAT <RETURN> from the command line. This will create (if not already on disk) the new file and allow entry of any new instructions (commands) you wish to include. Enter: "PROMPT \$P\$G" <Return> and then on the next line: "PATH C:\; C:\TP;" <RETURN>. Omit the quotes, of course, and you may want to include other directories in the "PATH" command. Now pressing "F2" will save your new creation to disk, and pressing "ALT-X" will exit Turbo.

If you are really dedicated, you might want to include, at the bottom of your AUTOEXEC.BAT file: cd\tp\turbo sortdir. This will change the directory (cd) to "\TP", invoke the Turbo compiler and load "Sortdir.pas". All without touching a key

upon boot-up.

After you have successfully copied this month's code listing and eliminated the inevitable typos, you may invoke "Sortdir" without any parameters by just entering the program name from the command line. Doing so will result in displaying a sorted directory of the default drive. Entering "Sortdir a:" will lead to a screen displaying the files occupying drive A's default directory (the root unless you have exited that drive while occupying another directory). The combinations are endless so I will leave it up to you to experiment with this program. It operates much like DOS' "Dir" command, but sorts the directories/files before displaying them.

Notice too, that it displays "More" at the bottom of the screen when more than twenty-three files occupy the directory being displayed. Pressing any key will fill the screen with the next "Page" (screen) of files.

Now that you know how to use Sortdir, let's look at the way variables are stored in the computer's memory. The "Var" statement in most Pascal programs is followed by a list of variables and their matching types. For instance, you might find "Var Row : Integer;" after the "uses" statement and before any actual code. The above statement directs the Turbo compiler to set aside two bytes of memory for "Row". This type of storage is called "static," since it does not change in size or memory location (address) after being allocated by the compiler. It is "global" in scope which means that it can be accessed by any part of the program. The same rules apply to other types of storage, from Bytes which take one byte of memory to arrays which might take enormous amounts of memory depending on their configuration.

What happens if the amount of data to be handled by the program is unknown, as in the case of a database program or an editor? One way to deal with this problem is to allocate more static memory than the program is likely to need, a very wasteful and inefficient method, not to mention the restrictions placed on such a program.

There is another, more efficient way to allocate memory as needed. On the fly, so to speak. Dynamic allocation is the name given to such a process and Turbo Pascal makes it quite easy. There is one aspect of this method which may cause you to wonder how the program knows where each block of memory resides. In other words, what is its address? Special attention must be given to this detail since, if we have no way to find it, the memory and its contents do us no good. Under Turbo Pascal, you can create "records" which use pointers, providing a means of allocating memory dynamically and creating a reference at the same time.

Sound complicated? Not at all. In fact, it is quite easy and straight forward. Let's

look at a code fragment from "SortDir", abbreviated for clarity. Under the "uses" statement we find:

```
Type String20 = String[20];
   LinkPtr = ^Link;
   Link = Record
     Words : String20;
     Next : LinkPtr;
end;
```

```
Var P, RunP : LinkPtr;
```

Here, we have a sample of a very powerful Turbo Pascal feature. We are creating a pointer (LinkPtr) to "Link," which in turn, is the name given to a Turbo Pascal "Record". Next, two pointers are declared: "P and RunP", which are of TYPE "LinkPtr". Now we can invoke another of Turbo Pascal's features: New(P);. This statement causes the allocation of enough memory to hold a 20 byte string (Words) and a pointer (Next) of TYPE LinkPtr. That process may sound quite innocent, but now we have a variable created on the fly (dynamic allocation) and a pointer (P) that may be used to access the record just created containing the string alluded to above, and a pointer (Next) which may be made to point to yet another record.

I know that sounds a bit complicated, but picture this: a series of memory locations, each containing the address of the next block of memory; thus forming a true chain. Any member of the chain may be visited by moving from one element to the next, using the built-in pointers to find the next element.

A graphic example might look like this: (P)ointer->[Record..(N)ext]->[Record..(N)ext]-> in a continuous chain for as many records as would be required for the application. Each of the above "Next" pointers would contain the address of the succeeding element. If the logic still seems a bit cloudy, hang on, we are going to utilize some of the above code in a real world example and run the debugger on it to "look inside". See Listing 1 for SORTDIR.PAS.

Space does not permit a thorough examination of the code in "SortDir" since many new routines are used and some of the code is a bit complex. However, we will have a chance to really get our feet wet in dynamic memory allocation, a very important topic. One that should be thoroughly scrutinized and that is our objective. Before perusing the actual code, notice the pointer variables that have been established under "Var".

One of the most important things to remember about pointers and linked lists, is that each block of memory can only be accessed through the use of a pointer. Given that fact, there must be a unique pointer for each block of memory. Equally important, there MUST be a separate pointer to the FIRST element in the list. Doublylinked lists require two unique pointers, but that is a topic for later discussion. Notice the procedure "Init(s : string20).

Here, we are establishing a pointer to the first element in our list using "New(Start)". Enough memory is allocated for a complete record and "Start" points to the memory location containing the first record. The pointer to the next record is cleared (Start^.Next:= nil) and a running pointer is made equal to "Start". The Boolean variable "First" is made equal to False (confirming that the initialization of our list has been completed). Next, the string containing the file info is read into P^.Words. If this is a directory entry, then a second pointer is made to equal "Start" (If Dir = True, then StartD:=Start). This procedure is used once, or at most, twice during the program execution. When the directory contains sub-directories as well as files, it is called twice; each time creating a pointer to either the first directory entry (first time) or to the first file listing (second call).

Now that the most crucial point is behind us, we can find the beginning of our list by using the pointer "StartD" if the first member is a directory entry or "Start" if it is not. We even have a Boolean variable telling us which case is True (Dir : Boolean).

OK, we have a good start, but the object to this whole exercise is to put everything in alphabetical order, so let's tackle that next.

The "Procedure Rest(S : String20)" is central to achieving a sorted list. There are three possible scenarios for the insertion of each item. It is either the first, somewhere between the first and the last, or the last item. Our procedure must provide a means of dealing with each case. Let's discuss the first case first. Our procedure is passed, using its parameter list, a string containing the file name of the directory entry. It is compared to the first item in the current list (P:=Start) and, this is very important, a running pointer is made equal to our pointer (RunP:=P). It can be compared to a bookmark since it is used to "hold our place" while other operations are being performed. If it (the string passed to "Rest") is found to be less than (If S < P^.Words [alphabetically]) item #1 on the list, it is inserted in front(New(P);P^.Words:=S;Start:=P;) making "Start" equal to the new pointer. Before leaving this operation, we must link the new entry to what had been item #1, but is now #2. We do this by employing our running pointer (P^.Next:=RunP), [no other pointer retains a reference to this block of memory] and then exiting the procedure using a goto (goto out;).

If our string, alluded to above, is not less than the first item on our list, it must be greater than item #1 (it can't be equal, the operating system won't permit it). The possibility remains that it may be greater than item #2, item #3 and so on. What to do? We must scan the list until we find the first entry that is greater than the current one, then insert our item in front of it. How do we scan a list that has no subscript as an

array would have? How do we know where the different items are? This is the point where our pointer logic comes to the rescue.

Each time a new member is added to our list, it is linked to a sibling by using the pointer built into the current member. We'll use our running pointer as a bookmark (RunP:=P;) and then advance to the next item (P:=P^.Next) until we find an entry greater than the current string (while S > P^.Words). When we find our much sought after item, "P" points to a record which is greater than our comparison string. You may be thinking "What now Sherlock?". "RunP" to the rescue, it is still pointing to the previous record. We will back up to "RunP" and use it to link with a new record. Let's use "MarkP" (see list of pointer variables) and make it equal to RunP (MarkP:=RunP;RunP:=P;). OK, now we have two "bookmarks", one smaller than the current string and one larger. We have only to create a new record and read the string into it (New(P);P^.Words:=S). Well, that's not really all, we must establish a link to its mates; MarkP^.Next:=P; P^.Next:=RunP; ought to do it.

You are probably thinking, "he forgot one small detail". "What if "S" (our current string) is larger than all the other items in the list"? This scenario is actually the easiest to deal with, since we merely have to tack the new entry on to the end of the existing list and make the "Next" pointer equal to "Nil" (providing a means of identifying the end of the list when traversing). Look closely at the listing to see how this is done (still within Procedure "Rest(S : string20)").

Ready for a little debugging session? OK, first eliminate all the typo's in this month's listing because it will be our "guinea pig" for Part 2 of the session. You'd like to use the debugger to eliminate the typo's? It's another case of overkill since Turbo's editor will do a good job in finding most of the Typo's, but "be my guest" in using it to find any remaining bugs, since I can't look over your shoulder.

We are actually going to employ the debugger as an educational tool rather than using it to find a program "bug". I left several areas in the listing with suggestions (see commented code) for changes to be made and then "watched" with the debugger. Let's fire up the debugger and watch what happens to our dynamic memory allocation in action. We will "look inside" the source code as entries to our list are processed.

Our first task will be to set up a testing environment which will provide an uncomplicated directory. We are going to create a sub-directory and copy a few files into it for testing, they can be removed (erased) after the completion of our test. Use the following commands: md\tp\test <Return>, copy \tp\sortdir.pas \tp\test <Return>. Copy 3 or 4 additional files,

possibly source code from the \tp directory, into this same sub-directory using the above command, but substituting the desired file names.

Assuming that Turbo Pascal resides in the \TP directory, use the following command from the root directory. "Cd\tp\test" <Return> "Turbo sortdir" <Return>. This command will change to the \TP\Test directory, invoke Turbo Pascal and load our program. Now press "Alt D" and observe the pull-down menu displayed on the upper right of your screen. You may scroll the highlighted bar up and down using the arrow keys. Move it down to the line: "Integrated debugging" and press <Return> several times while watching the mode toggle between on and off. Make sure that it is "on" and return back to the compiler by pressing "Esc". Run the compiler by pressing "Alt C", select destination: disk, and "build" from the pull-down menu and then press <Return>. After the compiler has written the "exe" version of "SortDir" to disk, we can place the cursor on any given line in the source code, press "F4" and then by using "F7" execute the code a line at a time, watching variables as we do so. One word of caution: The "watch window" is toggled on and off with "F5", toggle it to "on" if it is not visible, and then press "Alt B", select "Add watch" from the menu and then enter "P^.Words" in the window displayed. Using the same routine, add "RunP^.Words", "P^.Next", "S", and "P".

Ready for a bit of magic? While still in the Turbo Pascal editor, place the cursor within the "Procedure Rest(S : String20), on the line containing P:=Start;. Now press "F4" and then "F7" repeatedly, watching the values change in the "watch window" as the cursor traverses the procedure code.

You may want to single-step through the entire list of directory entries within the current sub-directory, watching "Justify" expand an entry and see how each list entry is processed. The possibilities are endless, but the bottom line is: an increased comprehension of how the code works. What a great tool!

Any time you wish to rerun the debugger, you have only to reposition the cursor on the line where you wish to begin to "Trace program execution", press "Ctrl-F2" followed by "F4" and repeat the above procedure.

I would suggest that you spend some time with the debugger on this program and any others that might hold some mysteries. This is a tool that I consider absolutely essential for the production of meaningful programs within a reasonable time frame. This does not even take into account its educational value.

See you soon.

Listing 1

```
Program SortDir;           { sorts directory }

uses dos, crt;

Type
String1 = String[1];
String20 = String[20];
String80 = String[80];
LinkPtr = *Link;
Link = Record
  Words : String20;
  Mon   : Integer;
  Size  : LongInt;
  Day   : Integer;
  Year  : Integer;
  Hour  : Integer;
  Min   : Integer;
  Next  : LinkPtr;
  DayHalf : String1;
  Dir   : Boolean;
end;

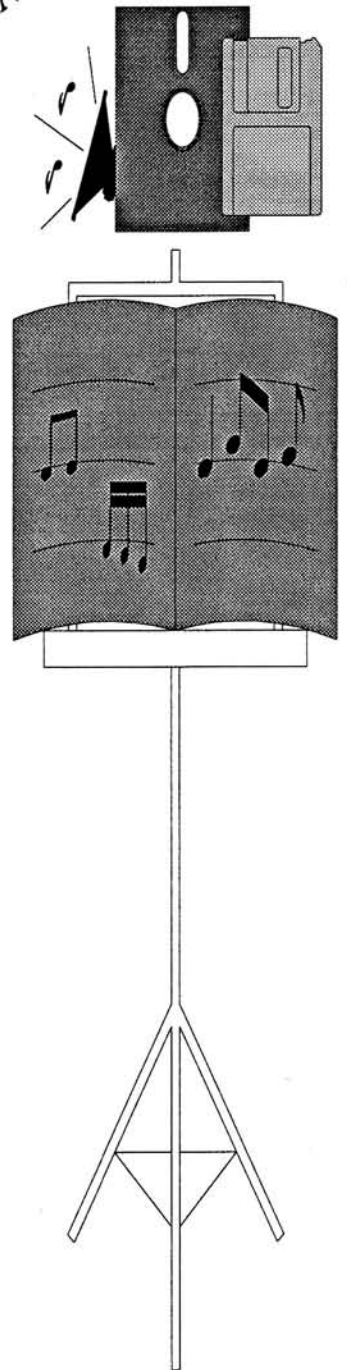
Var
StartD, MarkP, RunP, Start, P : LinkPtr;
Found, Dir, First : Boolean;
Attr, Year, Month, Day, DayOfWeek : Word;
FileInfo : SearchRec;
FileDate : DateTime;
Period, Path, CmdString : String80;
FileReference : File of byte;
Row, Count, RecNum : Integer;
NumKbytes : Longint;
D : DirStr;
E : ExtStr;
N : NameStr;
Scroll : Char;

Procedure Justify(Var S : String20); { ensures each screen display is }
  Var I, Len : Integer;             { the same length by tacking on }
  Begin                             { spaces, this routine could be }
    Len:=Length(S);                 { useful in many other programs }
    For I := 1 to 14 - Len do
      S:=S+' ';
    end;

Procedure Init(S : String20);
  Begin
    New(Start);Start^.Next:=Nil;    { creates a unique pointer to the }
    RunP:=start;First:=False;      { start of list, discriminates }
    Start^.Words:=S;P:=Start;      { between files and directories }
    if Dir = True then             { by creating a separate pointer for }
      StartD:=Start;              { each }
  End;

Procedure Rest(S : String20);
  Label out;
  Begin
    P:=Start;RunP:=P;              { Start at the beginning of list }
    if S < P^.Words then           { If current name < List name }
      Begin
        New(P);P^.Words:=S;        { Insert in front }
        Start:=P;
        If Dir = True then
          StartD:=P;
        P^.Next:=RunP;
        goto out;
      end
    else
      while (S > P^.Words)         { If current name > than list item }
        and (P^.Next <> Nil) do    { and not to the end yet }
        begin
          RunP:=P;P:=P^.Next;     { Advance through list }
        end;
      if (P^.Next = Nil) and (S > P^.Words) then
        begin
          RunP:=P;New(P);         { last item in list, append }
          P^.Words:=S;RunP^.Next:=P;
          P^.Next:=Nil;goto out;
        end
      else
        MarkP:=RunP;RunP:=P;
        new(P);P^.Words:=S;       { Insert in front of current }
        MarkP^.Next:=P;P^.Next:=RunP;
        out:
      end;
  end;
```

New Software Product!



The Electronic Clavier
P/N 885-6016

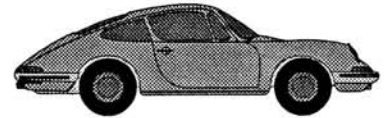
```

begin
First:=True;Dir:=False;          { initialize variables, clear screen }
count:=0;Found:=False;RecNum:=0;
clrscr;Path:=Paramstr(1);StartD:=Nil; { get path if on command line }
NumKbytes:=0;
Path := FExpand(Path);           { use Turbo library routine to expand }
if Path[Length(Path)] <> '\' then { path }
begin
Assign(Filereference, Path);
GetFAttr(Filereference, Attr);
if (DosError = 0) and (Attr and Directory <> 0) then
Path := Path + '\';
end;
FSplit(Path, D, N, E);
if N = '' then N := '*.*';       { if file name or extension not }
if E = '' then E := '*.*';       { specified, substitute *.* (wild cards) }
Path := D + N + E;              { re-assemble path }
gotoxy(34,12);write('Please wait..');
FindFirst(Path, AnyFile - VolumeID, FileInfo); { Turbo library routine }
while DosError = 0 do
begin
If (FileInfo.Attr = Directory)   { search disk for directories }
and (FileInfo.name[1] <> '.') then { omit the (.) directories }
begin
Dir:=True;Mark(P);              { establish location of top of heap }
if First = True then Init(FileInfo.name) { for later memory release }
else
Rest(FileInfo.name);           { create storage for file names }
P*.Dir:=True;Justify(P*.Words); { and insert in alphabetical order }
Found:=True;Inc(RecNum);
end; { end of if fileInfo }
FindNext(FileInfo);
end; { end of while }
First:=True;Dir:=False;         { omit for debug session }
FindFirst(Path, AnyFile - Directory - Hidden - VolumeID, FileInfo);
while DosError = 0 do           { omit volumeID for debug }
begin
UnPackTime( FileInfo.Time, FileInfo.Date); { repeat the above operation, }
If First = True then           { looking for files }
Init(FileInfo.Name) else Rest(FileInfo.name);
Justify(P*.Words);
assign(FileReference,D + FileInfo.Name);
Reset(FileReference);
P*.Mon:=Filedate.Month;P*.Day:=Filedate.Day;
P*.Size:=FileSize(FileReference);P*.Year:=FileDate.Year-1900;
P*.Hour:=Filedate.Hour;P*.Min:=FileDate.Min;P*.DayHalf:='a';
if (P*.hour >= 12) then P*.DayHalf:='p';
NumKbytes:=NumKbytes+P*.Size;P*.Dir:=False;
Found:=True;Inc(RecNum);
Close(Filereference);
FindNext( FileInfo);
end;
clrscr;writeln('Directory of ',Path); { display the path }
If StartD <> Nil then P:=StartD;
while P*.Dir = True do
begin
writeln(P*.Words,'<DIR>');       { display the subdirectories }
P:=P*.Next;Dec(RecNum);Inc(Count);
if count mod 23 = 0 then
begin
write('- More -');Scroll:=Readkey; { stop after 23, allow screen }
{ scroll }
end;
end;
P:=Start;                          { establish pointer to top of }
{ list of files }
begin
If(P*.Hour > 12) then P*.hour:=P*.Hour mod 12; { display files and }
Write(P*.Words,P*.Size:7,' ',P*.Mon:2,'-', { time/date info }
P*.Day:2,'-',P*.Year:2,' ',P*.hour:2,':');
If(P*.Min < 10) then Write('0');Writeln(P*.Min,P*.DayHalf);
P:=P*.Next;Dec(RecNum);Inc(Count);
if count mod 23 = 0 then
begin
row:=WhereY;
write('- More -');Scroll:=Readkey;
gotoxy(1,row);clreol;
end;
end;
if Found = True then begin
Write(' ',count,' Files using ', NumKbytes Div 1000,'K Bytes,');
Writeln(' ',DiskFree(Ord(Path[1])-64), ' bytes free on disk');
end
else
writeln('File(s) not found');Release(P);{release memory back to heap}
end.

```

end.

*



MOVING?
Don't miss a single issue of REMark!

Please let us know 3-4 weeks
before you move. Call
(616) 982-3463 or write:

Zenith Data Systems
P.O. Box 217
Benton Harbor, MI 48023-0217

Is lack of computer knowledge
holding you up?

"Powering Up"
could be your best protection!



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

WordPerfect

Jerry Klinke
2769 W. Glenlord
Stevensville, MI 49127

WordPerfect

WordPerfect

Shrinking WordPerfect 5.1 to Fit on a WordPerfect WordPerfect MinisPort ZL-2

Most people purchase the MinisPort for word processing, and the software of choice is WordPerfect 5.1. But looking at how much hard disk space is occupied when installed on a desktop computer (about 2.5 MB), one would think it would be impossible to put it on the small RAM disk (Silicon Disk Drive, or SDD) in the ZL-2. In this article, you will learn what files you need, what they do, and how to best use WordPerfect with the MinisPort. For brevity (and because WordPerfect is such a powerful program), I will assume that most people reading this article are at least familiar with the basic operation of the program. Most of what is covered here can also be applied to the ZL-1. Some specifics are covered at the end just for single 2" floppy use and the 1 MB limitations associated with the ZL-1.

Since WordPerfect Installation disks are not available in a 2" format and the installation program does not give you much control on what files are installed, I would recommend first installing the program on a computer with a hard drive and configuring the program for the printer(s) that you use. Then, using the FastWire Lynx transfer program and cable (that comes with the MinisPort), transfer only the files that you need. But remember, WordPerfect is copyright protected. You cannot use it on both computers unless you own two copies of the software or have made other arrangements with WordPerfect Corporation. So, when the file transfers are complete, remember to remove all the WordPerfect files installed on the hard drive unit.

Basic File Needed

You actually only need two files to use WordPerfect!

With the two files shown in Figure 1, this program will operate fine, but you will not be able to use some of the "bells & whistles" that WordPerfect is noted for. Although you can use a majority of the basic features, including the pull-down menus and the graphics capabilities, you can't print, view, use help or the spell checker or thesaurus, edit macros, or create complex equations. But individual use varies from person to person. By starting with these two basic files, I will describe how you can "build" your own customized version of WordPerfect.

Filename	Size
WP.EXE	218112
WP.FIL	572569

Figure 1

Add a Printer

Most people like to be able to print directly from the MinisPort, and the additional files needed do not occupy much space. The two "basic" files only require about 790 KB, leaving about 500 KB free on the SDD (considering that all free memory was allocated to the D: drive). But before you pack this drive full, keep in mind that WordPerfect needs extra disk space when it operates. I will briefly explain this later, but use a conservative approach when choosing the files you wish to put on the SDD. There are two types of printer files used — the *.ALL files that contain many different printers, and the *.PRS files that are created from the *.ALL files. During the WP installation, you should have selected the printer(s) that you use, creating a file with a *.PRS extension. This file (or files) is all that you need to perform printing. For

example, I use an Epson 2500 at home and an HP Series II LaserJet at work, so I only need the *.PRS files as shown in Figure 2.

Filename	Size
EPLQ2500.PRS	12291
HPLASEII.PRS	40155

Figure 2

These files are referred to as "Printer Resource" files. They contain all the information needed by WordPerfect to use every feature and font available with that printer. STANDARD.PRS (2 KB) is a generic Printer Resource file that will let you print (text, no graphics) on most dot matrix printers in draft quality only. It was installed with the rest of the WP files. It is not my first choice of a Printer Resource file, but may be all you need.

WordPerfect has over 1500 characters available, but printing and viewing these characters graphically requires another file. You only need one of the two files shown in Figure 3.

Filename	Size
WPSMALL.DRS	48357
WP.DRS	489586

Figure 3

WP.DRS is more than 10 times the size of WPSMALL.DRS. It would seem like something that large will perform some awesome feats — and it does! WP.DRS allows you to print (and view) any of the 1500+ characters that WordPerfect can generate, as long as you have a graphics printer. WPSMALL.DRS will allow you to print (and view) only those character sets available as listed in your printer manual. These character sets are sufficient for most people. If you really need others, like the icons and large

math symbols, you can still enter them in your document and they will appear as a small square dot. In order to print these, you will have to transfer your saved document to a computer with WordPerfect that is using the WP.DRS file. You can then print the file from that computer.

Graphics

The two basic files contain everything that is needed to create, edit, and print graphics in the document, except if you create equations. You must then include WP.QRS with the other files.

Add a Video Driver

If you want to use the View Document feature, you need to have both the *.PRS and the WPSMALL.DRS files, as well as a Video Driver. These are noted by their *.VRS extension. The Video Driver that works with the MinisPort is shown in Figure 4.

Filename	Size
STANDARD.VRS	29833

Figure 4

There are other *.VRS files that are smaller in size, but none of them will work with the MinisPort.

Need to Edit Macros?

If you are a "Macro Maniac" and need to alter previously saved macros, you need to include one or both of the files listed in Figure 5.

Filename	Size
KEYS.MRS	4800
WP.MRS	5912

Figure 5

The first listing is for keyboard macros. The second one is for regular macros.

If you use some of the keyboard macros provided by WordPerfect, you will need to copy the one you use. It will end with a *.WPK extension.

There are also pre-defined "regular" macros. These end with an extension of *.WPM. Copy only the ones you use regularly. Or keep them on a separate disk and copy them to your SDD as they are needed.

Other Files

Speller... Depending on how many files are on the SDD, this may fit. If not, it can be installed and run from a second floppy disk. The two files that you will need are shown in Figure 6.

Filename	Size
WP{WP}.SPW	13126
WP{WP}US.LEX	363220

Figure 6

Thesaurus... This one is also a large file and may not fit on the SDD. Use a second floppy for the single file shown in Figure 7.

Filename	Size
WP{WP}US.TH5	358427

Figure 7

The Mouse... If you are one of those people who just can't leave home without one of these rodents, you will need a mouse driver. You could use MOUSE.COM that comes with the mouse, but this file requires about 27 KB of disk space. WordPerfect includes a driver that works with most mice. The file is called STANDARD.IRS. It is only a 5 KB file and will work with most popular mice. This driver works within WordPerfect, so just select your mouse with setup (Shift-F1,1,1). You do not need to put it in your AUTOEXEC.BAT or CONFIG.SYS files.

Help... The help features provided with WordPerfect are a great asset; however, the file needed for this is fairly large. The file name is WPHELP.FIL and is about 182 KB in size. If you don't need on-line help, leave this file out.

Cursor Problems?

Some people have trouble finding the cursor with WordPerfect on LCD screens. There is a file called CURSOR.COM that is a slick cursor control program. Although you can choose from about 40 different types and sizes of cursors, most people use the large-sized cursor. Zenith Data Systems also has a fix for this "lost" cursor problem. It is installed on the C: drive and does not take any valuable space on the SDD. To use it, enter the statement in Figure 8 at the DOS prompt, or include it in your AUTOEXEC.BAT file.

C:\MACHINE CURSOR FULL

Figure 8

Improving Cursor Visibility

In order to see the cursor when you use the "reveal codes" feature, you must change the screen colors in the setup portion on WordPerfect.

- Set the display
- Go to "setup" (Shift-F1,2,2) and select:

- IBM CGA (& compatibles)

Then select the following at the next screen:

- CGA 640x200

While still in the SETUP:DISPLAY option menu, go to the Colors/Fonts/Attributes and change only the "Blocked" line as shown in Figure 9.

Attribute	Foregrd	Backgrd
Blocked	B	H

Figure 9

You should go through the other setup screens and configure the program to your

specifications. (Refer to the WordPerfect manual for more information.) You will find most of the default settings satisfactory for the MinisPort.

Typical Setups

Now that you know what the files do, you can "build your own program," customizing it to your needs. I would recommend keeping all the WordPerfect files in a subdirectory on the D: drive under D:\WP51 and keeping the documents that you create on A: drive.

The following are examples of various configurations.

Example 1. The configuration shown in Figure 10 will allow you to use two printers, the View Document features, and print only those characters that the printer has available. It will not create equations, edit macros, or perform spell checking.

WP{WP}.SET	3k
EPLQ2500.PRS	13k
STANDARD.VRS	30k
HPLASEII.PRS	40k
WPSMALL.DRS	48k
WP.EXE	213k
WP.FIL	560k
7 files, approximately 907k	

Figure 10

If you are wondering about the first file in Figure 10 (WP{WP}.SET), this is created the first time you use WordPerfect. It contains all of your setup information in regards to printer, drives, etc.

Example 2. The configuration shown in Figure 11 will allow you to use two printers, the View Document feature, perform spell checking, and will print only those characters that the printer has available. It will not create equations or edit macros.

WP{WP}.SET	3k
EPLQ2500.PRS	13k
WP{WP}.SPW	13k
STANDARD.VRS	30k
HPLASEII.PRS	40k
WPSMALL.DRS	48k
WP.EXE	213k
WP{WP}US.LEX	355k
WP.FIL	560k
9 files, approximately 1275k	

Figure 11

Example 3. The configuration shown in Figure 12 will allow you to use one printer, the View Document feature, create equations, and will print any character listed in the WordPerfect Manual. It will not perform spell checking or edit macros.

Example 4. The configuration shown in Figure 13 will allow you to use one printer (text, no graphics). It will not perform the View Document feature, create equations, print any characters or graphics, edit macros, or perform spell checking.

Example 5. The configuration shown in Figure 14 will allow you to use 1 printer,

Continued on Page 48



Modems

Part 5

Winding Through the Protocol Forest

Robert C. Brenner, MSEE, MSSM
1991 Brenner Information Group
9282 Samantha Court
San Diego, CA 92129

Webster's New Collegiate Dictionary defines a protocol as a "code prescribing strict adherence to correct etiquette and precedence." The distinction between standards and protocols is about as clear as mud. However, we can agree that a protocol is a set of rules, a formal set of agreed conventions covering format and timing for the exchange of data or information files by two separated communication systems.

This article focuses primarily on the most common data handling and transfer protocols between computer/modem systems.

Generally, protocols can be partitioned into software and hardware rules. Software protocols handle error detection and correction using utilities that the user runs as part of a communications program. The comm program in the computer at each end of a transmission link must use the same protocol for optimum data transfer. Increasing the volume of data increases the risk of data transmission errors. When errors are detected, the data is retransmitted. Retransmitting a complete file whenever an error is detected is wasteful, so protocols were developed to identify and correct transmission errors before, during, or after transfer. In this way, only the part of the data containing the error is retransmitted.

Hardware protocols act on the data as it is being moved into and out of the modem. These protocols support "on-the-fly" error detection, data compression and format translation. Many communication standards are also referred to as protocols. Protocols are being implemented in hardware (firmware) to provide improved performance. The following protocol information was discovered during research on this series.

Protocols insure an orderly transfer of information. Advanced protocols detect and handle errors during transfer. There are also protocols that control data transfer asynchronously or synchronously. A file transfer protocol defines how data will be packaged, how much data will be sent in a unit, how often data will be sent, how errors will be detected, how data receipt will be acknowledged, and how retransmission of data will occur when an error is detected.

File Transfer Protocols

The following file transfer protocols use character transfer formats for sending and receiving data.

XON/XOFF is an asynchronous start-stop half-duplex flow control software protocol that's useful when exchanging files with different type computer systems. The receiver captures data in a memory buffer and sends an XOFF to the sending computer when its buffer is nearly full. Transmission stops and the buffer is written to disk. Then, XON is passed to the sending unit and transfer begins again. XON/XOFF is the simplest form of protocol.

ASCII is a low-level protocol format used for quickly transferring text files. This protocol has no error checking and no special features. But it's simple and fast. The ASCII protocol is also called Text protocol.

Keith Petersen adapted his Modem program for unattended operation and called it XModem. As public domain software, XModem quickly became a de facto file transfer standard. XModem has several deficiencies, including a short block length (128 bytes), a single byte checksum for detecting errors, and the ability to only handle a single file at a time. This simple transfer process was intended for "short

haul" file transfers between computers located in the same general area. It was also designed when 300 baud was the standard. XModem introduces overhead (wasted time costs) to data exchange. As bulletin board systems (BBSs) proliferated and users began sharing files across hundreds of miles, XModem was found inadequate. This prompted various upgrade mods to the original XModem protocol. Nevertheless, XModem is the de facto standard file transfer protocol for uploading and downloading data so most programs handle it. Like the original Modem program, XModem sends data in packets of 128 bytes and uses a checksum to determine transfer accuracy. Even though other protocols provide better error correction, XModem is widely accepted and used. XModem is also known as Christensen, XModem Checksum, and XModem 128.

Most communications software can operate under the XModem file transfer rules. Running XModem with an automatic error correcting modem isn't recommended because forcing data through two error correcting protocols can reduce performance. Various upgrades and extensions to XModem evolved to overcome its shortcomings, particularly the one-byte checksum and to enable this protocol to work with advanced technology.

XModem CRC uses two-byte cyclical redundancy check in the data exchange. If a modem tries to communicate with another modem using XModem CRC, and one modem cannot understand this protocol, both modems will revert to standard XModem and use checksum error checking.

XModem CRC Batch is an upgrade to XModem. It uses a two-byte cyclical redundancy check (CRC) value in the exchange.

CRC testing can detect 99.99 percent of transmission errors.

Kermit is used for data transfer between micro and mainframe computers. Some mainframes have difficulty handling control codes and use 7-bit characters instead of the 8-bit ASCII codes common to microcomputers. The file transfer protocol used with some mainframes must perform the bit conversion. Frank da Cruz, Bill Catchings and other computer scientists at Columbia University developed Kermit as a public domain protocol to perform these conversions and enable file transfers between microcomputers and mainframe systems. With authorization by the late Jim Henson, Kermit was named after the lovable green frog in the Muppets group.

Kermit is a half-duplex, character-oriented protocol that handles buffering, character set conversion, duplex, parity, and other link problems in linking micro-, mini- and mainframe computers. A special version of Kermit exists for every major mainframe and microcomputer system, including Macintosh, PC- and MS-DOS systems. Everything is sent in packets including the control sequences for remote systems. Kermit encodes binary data into 7- or 8-bit character ASCII depending on the direction of transfer. The protocol builds frames of about 96 bytes and incorporates one of three checksum techniques.

Kermit normally uses 96 of the 256 characters specified in the ASCII character set. After collecting everything into packets, this protocol program assigns a special "prefixing" for control characters and high-bit characters. Kermit allows multiple file transfers and can work through almost any environment in a master/slave configuration. But it has no provision for sending file type information to a receiver. Therefore, Kermit is normally used only for text file transfer. Although slower than XModem, it is the choice for many research networks (e.g., Bitnet and Usenet). Kermit has no file mode. It uses a batch transfer mode only, even if the batch is actually a single file.

Local users groups and bulletin boards offer microcomputer versions of Kermit. The PC versions of Kermit emulate the Heath/Zenith H-19 terminal recognized by most UNIX systems.

Chuck Forsberg cleaned up the implementation of batch mode in XModem and implemented other enhancements into a new protocol called YModem. YModem is an integrated part of his public domain communications program YAM ("Yet Another Modem"). Besides cleaning up batch mode, YModem adds features so a standard data packet can include the filename, CRC, and machine specific information. Although based on XModem, YModem is not obligated to support XModem and its derivatives. It operates asynchronously with 8-bit data and collects data into 128- or 1Kbyte blocks (packets). Obviously, the 1K

blocks are more efficient. YModem still uses the acknowledge (ACK), not acknowledge (NAK) handshaking of XModem, thereby directly affecting throughput. YModem is best used on lines with modems that contain hardware error detection and correction.

YModem-G is a streaming protocol that eliminates the ACK/NAK and sends data all at once. About the time YModem-G was coming on the scene, microcomputers were connecting to on-line network services and transferring data in packets of varying lengths. YModem-G was written for high-speed modems with built-in error correction circuitry. This protocol doesn't support error recovery. It causes the sending unit to finish transmitting the entire file before checking for errors. If an error is detected, the session is aborted or repeated. YModem-G is limited to hard-wired configurations or in links where the hardware handles error correction. YModem-G is ideal for use with error-correcting modems because it doesn't introduce error correction conflict.

ZModem is a full duplex streaming protocol with error correction. Chuck Forsberg designed ZModem in 1987 with funding from Telenet. ZModem incorporates error detection and correction in packet switched networks. The advanced error handling in ZModem lets users transfer error-checked files in propagation delay environments, such as between microcomputers and mainframe computers. It is also used in packet-switched networks, such as Telenet or Tymnet, even when the telephone lines are poor. It was originally written for data transfer over satellite links.

The ZModem protocol was placed into the public domain. It collects the exchange data into 1KB packets and streams the packets out without waiting for an ACK signal return from the receiver. The sending and receiving ends are not synchronized. Satellites in the transmission path introduce inherent delays in the system. If a computer must wait for an ACK before sending the next packet, the whole transfer slows down. By not expecting an ACK, ZModem overcomes this limitation. XModem and YModem are affected by network propagation delays.

If an error does occur during a ZModem transfer, a NAK is returned leaving it up to the sending system to determine which packets to resend. Retransmission usually begins at the point where the bad block was detected. ZModem speeds up throughput when exchanging data over packet-switched networks where the session-level protocols associated with the network can add more delays in file transfer.

ZModem protects data transfers using 16- or 32-bit CRC values that are transmitted with the data. By increasing the number of bits in the CRC polynomial, undetected

errors can be reduced by at least five orders of magnitude. This protocol also contains a special security mechanism to guard against the "Trojan Horse" message that may try to mimic legitimate download commands.

The ZModem protocol contains a "checkpoint restart" feature that is popular with many users. If a modem is inadvertently disconnected from the telephone line, and a session is "knocked off line" in the middle of a file transfer, simply reconnecting the modem and redialing to re-establish the connection lets you continue file transfer at the point of interruption.

Other nice features of ZModem include automatic download, immediate file transfer without the ten second delay common to XModem transfers, and automatic protocol step-down to YModem if the receiving end cannot support ZModem. This protocol provides superior performance over XModem and YModem. ZModem is maintained in the public domain by the originator via Omen Technology,

W/XModem is a windowed protocol for packet-switched networks. Peter Boswell of People/Link created Window XModem (W/XModem) to improve throughput by avoiding the start-stop in the ACK/NAK process of XModem. It combines four data packets into a window. Then each window is sent without handshaking between the packets inside the window. The packets are expected to be without error. If a NAK occurs, and the transmitting computer still contains the failed packet in its send buffer, it resends it and proceeds. The W/XModem protocol can fail when the telephone line gets noisy. It works best on noise-free network lines.

XModem Sliding Windows is a batch protocol designed to let users transfer several files at once. Although powerful, batch protocols have been superseded by "archiving" programs that collect several related files into a single compressed "archive" file that is then saved on disk. Archiving programs eliminate the need for batch-file transfers. XModem Sliding Windows is similar to YModem Batch.

SuperKermit is a windowed version of Kermit. SuperKermit is useful for communicating over links that include satellite delay. The Source and Columbia University collaborated to create SuperKermit for sending up to 31 packets in a window. Each packet frame contains up to 90 bytes of data. Users claim that SuperKermit is about as fast as YModem, because it has the ability to accept late acknowledgements in a sliding window of time.

Error Correcting Protocols

Another area where protocols are important is in error correction. With information requirements now pushing the 3,100 Hz bandwidth limit of standard telephone lines, only innovative data compression

algorithms can allow faster data transfer rates using analog tone telephone technology. The two primary protocols for error correction are the Microcom Networking Protocol (MNP) and the Link Access Protocol (LAP) for modems.

Microcom Networking Protocol (MNP)

In 1982, modem manufacturer Microcom of Norwood, Massachusetts developed the Microcom Networking Protocol (MNP). MNP is based on byte-oriented data exchanges at half-duplex using dumb asynchronous modems.

A three step procedure establishes linkup. The sender issues a link request to a receiving unit. The receiver responds by issuing its own link request. The original caller then issues a link acknowledgement and transfer begins. Each message is exchanged using byte-oriented data packets comprised of an information header (packet function type, message sequence number, and 16-bit CRC code) and the data itself.

In 1984, Microcom moved this link layer protocol into its modem hardware and licensed the technology. The MNP design was so innovative that competing manufacturers swallowed their pride and integrated MNP into their own modems. MNP quickly became a standard solution for hardware-based error control, freeing communication software from data disruptions caused by line quality distortions.

The broad acceptance of MNP as a de facto standard is seen in the many modems that publicize MNP capability. Modems that implement MNP can still communicate with modems lacking MNP capability. MNP has been accepted by CCITT as suitable for providing error correction and file transfer support. In faster modems, trellis code modulation is typically the first line of defense for fixing transmission errors. If it fails, MNP takes over and asks the originating modem to resend the offending data packet. If there are still too many errors, the sending modem is instructed to re-organize the data into smaller packets.

Microcom has introduced 10 classes of their MNP protocol, extending the original error control MNP Class 1 to include data compression and extended services, such as full duplex simulation, and adverse channel enhancements. Class 1 is a simple protocol for asynchronous byte-oriented half-duplex data transmissions. Classes 2 through 10 have been implemented in hardware using ROMs designed into smart modems. The MNP error-correcting protocols resend a corrupt packet and all subsequent packets, reducing the amount of retransmission required. MNP Classes 1-4 are windowed protocols that send several packets in a burst without waiting for acknowledgement. Classes 2-4 have error checking protocols. Classes 2 and 3 are used for many low-speed transfers. Class 5 covers error correction and data compression.

Class 7 enhances the Class 5 data compression protocol. Classes 9 and 10 are enhancements to optimize data transfer speed.

Classes 1 through 4 have been adopted by the CCITT as standard for error correction. These classes have been released to the public domain by Microcom. The Microcom Networking Protocol classes are compared in Table 1.

Link Access Protocol (LAP)

The link access error correcting protocols were defined by the CCITT in their X.25 and ISDN recommendations. LAP-B and LAP-D are "windowed" packet protocols that can send several packets in a burst without waiting for an acknowledgement from the receiver. LAP-B (link access protocol-balanced) is a host-to-node datacomm protocol contained in the CCITT's X.25 multi-level protocol. It establishes balanced connection-oriented communications for a one-to-one link over a dedicated circuit between two parties. LAP-D (link access protocol—digital) defines the link protocol for Integrated Services Digital Network (ISDN) systems.

LAP-M (link access protocol—modem) is an error-correcting protocol promoted by Hayes and based on LAP-B and LAP-D. LAP-M is the primary protocol of the V.42 CCITT standard. Based on a long-standing synchronous protocol called "High-Level Data Link Control" (HDLC), LAP-M supports digital phone networks such as ISDN. It's a windowed packet protocol similar to MNP1-4.

LAP-M features automatic feature negotiation, a powerful way to determine and establish the best error control method between two modems.

CompuServe A, B, and B+ are network protocols for use with CompuServe on-line service. CompuServe A was originally developed to allow binary and hexadecimal file 7-bit ASCII transfers between microcomputers and mainframes containing the service. CompuServe B is in the public domain. It is optimized for CompuServe block sizes and handles delays common to mainframes operating with packet-switching networks. CompuServe B was created to support text file transfers with full-color video graphics. An advanced version of this protocol is called CompuServe B+.

Protocol Spoofing

What do you do when two modems both use XModem, but one handles data in

MNP Class

2 - 4
5 and 7

6 (full duplex simulation)
9 (protocol throughput enhancement)
10 (adverse channel enhancements)

Service

Error Control
Data Compression,
Error Control
Extended Services
Extended Services
Extended Services

Table 1
The 10 classes of the Microcom Networking Protocol.

128 byte blocks and the other packs 1024 (1K) bytes into a single block? Some modems incorporate a technique called "protocol spoofing" that makes the sending modem think data has been accepted without detected error when each block it sends is just being queued at the receiving end in a buffer. Each time a "mini-block" comes in, the receiving modem sends an acknowledgement initiating the sending of the next mini-block. If any error is detected, the receiving modem returns a retransmit request. Once the block size of the receiving modem has been matched, it collects the "maxi-block" into one sequence and passes it out its serial port into the connected computer or terminal.

Protocol spoofing has successfully increased throughput in some modem designs.

Network Communication Protocols

With the proliferation of network communications, there came the birth of several new software interface protocols. By pooling a number of modems on a single LAN, all the resources connected in the LAN can share and access any modem. To enable multiple modem sharing, interface protocols have been developed. The most prevalent modem pooling interface protocols are INT 14 and NASI.

INT 14 is a software interface technique that lets communications software installed in a workstation operate across a LAN instead of accessing the PC's serial ports directly. INT 14 uses the PC BIOS serial I/O service request interrupt (INT14). At least seven different INT 14 interfaces have been designed. Two come from IBM; another comes from Concept Development Systems. Comm software developers may or may not adhere to these three INT 14 implementations. This makes INT 14 an uncertain choice for a modem-pooling interface (unless every workstation on the LAN runs the same network communications program).

Besides a confusing array of implementations, the INT 14 interface doesn't make a provision for naming modems on the network. The modems and serial lines are distributed on a first-come, first-served basis. This makes it difficult to mix 1200, 2400, and 9600 bps modems as shared resources on a single LAN.

NASI/NCSI is a modem pooling inter-

face based on the original 6B communications services protocol defined by Ungermann-Bass. Network Products designed and licenses its NetWare asynchronous Services Interface (NASI) as a standard for modem pooling. Network Products also sells products that implement its own version called Network Communications Services Interface (NCSI). The NASI/

NCSI protocol is clearly defined. Although custom enhancements are implemented by Network Products, the commercial NCSI is virtually identical to NASI. NASI is supported on NetBIOS-compatible and NetWare networks. The NASI/NCSI protocol includes resource naming. PC Week calls it the modem-pooling protocol of choice.

As you can see, many protocols exist in the world of telecommunications. An in-depth discussion of protocols and other modem-related subjects can be found in the book *Modems Made Easy*. (This article was paraphrased from *Modems Made Easy*, Brenner Information Group, 9282 Samantha Court, San Diego, CA 92129 \$19.95.) *

Continued from Page 44

```
WP{WP} .SET      3k
EPLQ2500.PRS    13k
WP .QRS         17k
STANDARD.VRS   30k
WP .EXE        213k
WP .DRS        479k
WP .FIL        560k
7 files, approximately 1315k
```

Figure 12

```
WP{WP} .SET      3k
STANDARD.VRS   30k
WP .EXE        213k
WP .FIL        560k
4 files, approximately 777k
```

Figure 13

the View Document feature, print only those characters that the printer has available, and provide on-line help. It will not create equations, edit macros, or perform spell checking.

Disk Full... Don't panic if you see this message when using the program! When you start WordPerfect it automatically cre-

```
WP{WP} .SET      3k
EPLQ2500.PRS    13k
STANDARD.VRS   30k
WPSMALL.DRS    48k
WPHELP .FIL    182k
WP .EXE        213k
WP .FIL        560k
7 files, approximately 1047k
```

Figure 14

ates temporary and overflow files on the SDD to put your document in. These files perform various sorting, viewing, and printing functions. As a "rough" estimate, you will need about 2 times the document size of free space on the SDD to be able to work with it.

If you use a lot of the features listed and your SDD (D: drive) is full, there is still hope. All of the temporary and overflow files listed above can be redirected to the A: drive instead. This will affect performance since the floppy is slower than the RAM drive. Utilize this option by starting WordPerfect with this command:

WP/D:A:

When you save a document it re-names the old file with a *.BK! on the same drive as the original file. If you exit WordPerfect properly, this file is automatically deleted. If you don't have enough free disk space, you can cancel this feature by starting WordPerfect with the following command:

WP/NB

Printer Tips

When you print a document that is on the screen, it is "background" printed. This is accomplished by creating temporary files. If there isn't enough disk space, you won't be able to print. Another way of printing is to "Print Document on Disk" (option 3 from the print screen). Even with this option, it may not print if you saved the document

with the Fast Save option (the default setting). When a document is Fast Saved, it is not formatted; WordPerfect needs to retrieve the document in the background (not to the screen), move through the document to reformat it, and then send it to the printer. You can change this Fast Save option in setup (Shift-F1,3,5,No). This will affect the speed that documents are saved, but allows you to print certain large documents without requiring extra SDD space.

ZL-1 Use

Many of the items explained in this article can be applied to the ZL-1, except that not enough room exists on the SDD to install the program files. If you have an external drive, you can save your documents there or run the speller. Otherwise, you will have to save to the SDD (D: drive).

If your documents are small (less than 100 KB), you can direct the temporary and overflow files to this drive. This improves performance and conserves battery power. The WordPerfect program files must be installed on floppy disks (as shown below) and run from the A: drive. To utilize the "redirect" feature when starting WordPerfect, follow the instructions below.

If using D: drive to store your data, start WordPerfect from this drive by entering:

A:WP/D:D:

Put the files on disk as shown. Start WordPerfect on Disk 1. After a minute or so the program will instruct you to insert Disk 2. You should not have to reinsert the first disk.

Disk 1

WP.EXE

Disk 2

```
WP.FIL
WPSMALL.DRS
STANDARD.VRS
.PRS
.MRS
.WPK
.WPM
```

Conclusion

There are many different ways to install WordPerfect on your MinisPort. I have given you some essentials on the important areas, but like most programs, by digging into the manual, you can find many more options and ways to do things. There are numerous startup, printer, and macro options that I did not address here. *

"What Did I Change?"

Answer that question by using Directory Compare. Compare date/size of files in any two directories. Public Domain (source included). Send only \$5.00 shipping & handling. For PC or Z100 (specify diskette size)

"Can I Improve My Screen Prints?"

Yes - with EGAD Graphics & Text Screen Print PC's and now also for the Z-100 series.

- Prints *IN COLOR* on color printers, or uses black, white and *six gray tones* on most other printers
- Print *any part* of the screen - crop box pops up when Shift-PrtSc (PC) or Shift-F12 (Z-100) pressed; use arrow keys to select region.
- Enlarge graphics 1-4 times
- Supports Super VGA, VGA, EGA, and CGA
- SET program selects printer colors, other options.
- Supports most dot matrix, laser, and ink jet printers (including Epson, NEC-8023, MPI, Okidata, HP, etc.)

EGAD for PC's, Order # 270 \$35.00 postpaid. Specify 3.5" or 5.25" diskette.

EGAD for Z-100, Order # 271 \$35.00 postpaid.

LS Software (formerly Lindley Systems)
8139 E. Mawson Rd., Mesa AZ 85207
(602) 380-9175. Call/Write for Free Catalog

A decorative border of stylized yellow and orange flames surrounds the central text. The flames are outlined in red and have a jagged, flame-like appearance.

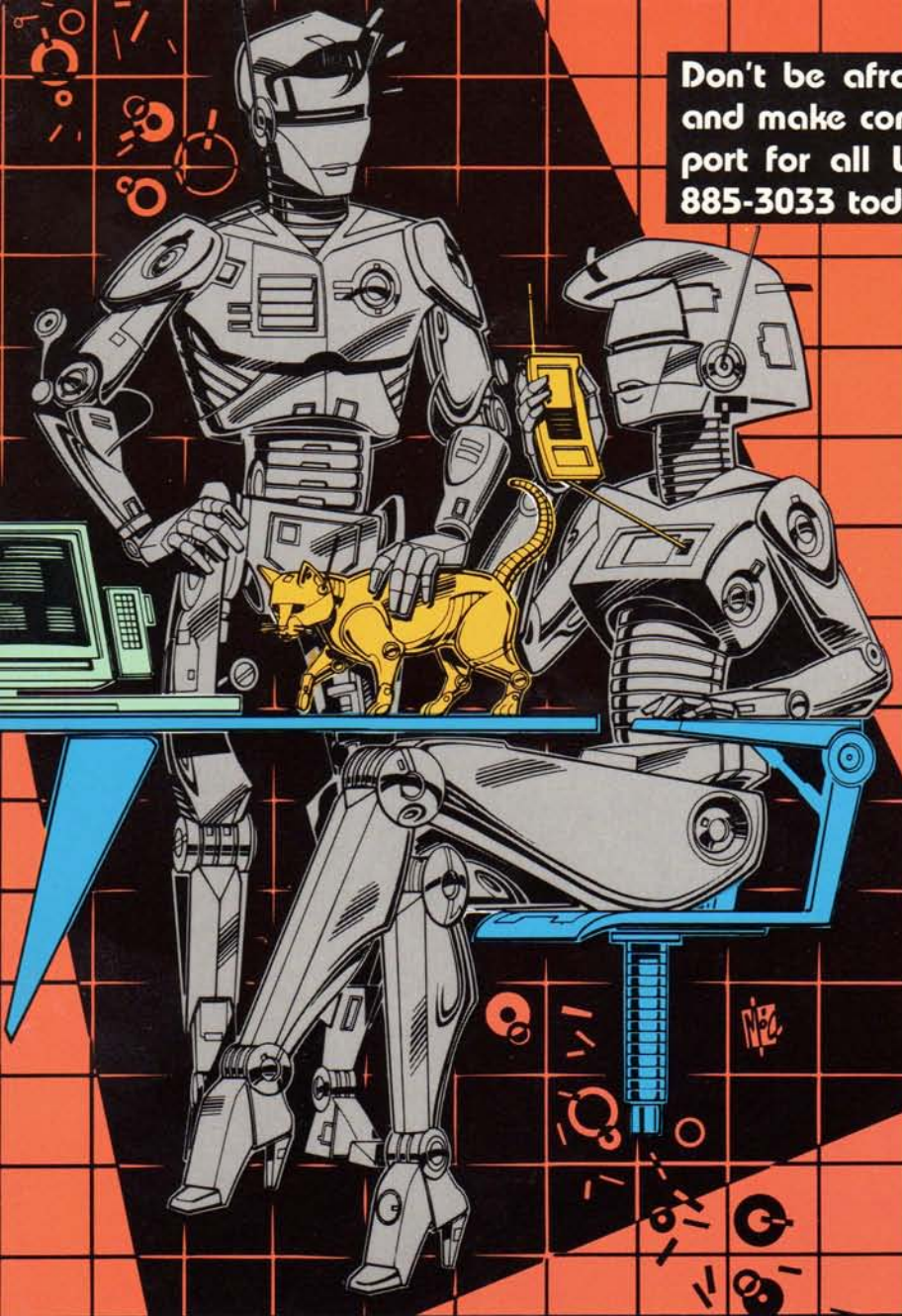
HADES II

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

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

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

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



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

Storage Buffer = 524288 Bytes
Storage Buffer Usage = 0 Bytes

Select Message (0-0), (F1) To List, Anything Else To Abort --> _
F1-Hlp F2-Msg F3-Bufr F4-Sav F5-Snd F6-Cfg F7-Clr F8-Print F9-Exit CM
```

```
HUGMCP Configuration Help #1
1- This Function Allows The Read Data To Be Changed, Depending Upon Which
      Mode It's In. Normally It Will Be Set To Either 100, 1200, Or
      2400 Baud. Selecting 0 To A Max. Will Allow Higher Baud Rates.

2- This Function Allows You To Change The Word Parity. Normally you
      should leave "No Parity". That Is Acceptable By Most Serial Buses,
      And It Is Also Necessary For XMODEM Protocol To Work Properly.

3- This Function Allows The Changing Of The Word Length. Normally The
      length should be set to 8 bits. This Value Is Acceptable By Most
      Serial Buses, And Is Necessary For XMODEM Protocol To Work Properly.

4- This Selection Allows You To Enter Messages Which Can Be Automatically
      sent With The (F2) Msg. (0-0) To (F1) Character Messages Can Be Entered.
      Selection (0) Is Special. It Should Contain Your Computer's Name
      And Forwarded Selection (0) To Allow Special. This Selection Can Auto-
      matically Be set With This Program If First Executed By Initiating The
      Program With The Setup.

Type (F6) For More Help, Anything Else To Configure.
F1-Hlp F2-Msg F3-Bufr F4-Sav F5-Snd F6-Cfg F7-Clr F8-Print F9-Exit CM
```

```
HUGMCP Configuration Menu:
# -- Modify Baud Rate
# -- Modify Parity Type
# -- Modify Word Length
# -- Modify Or Add Auto-Messages
# -- Miscellaneous Functions:
# -- Change Screen Color Assignments
# -- Display Current Configuration
# -- Make Changes Permanent

Select 0-5, (F1) For Help, Anything Else To Quit --> _

Baud Rate: 19200
Parity: NONE
Word Length: 8
Bauds: FULL
Response To Keyboard Disable: NO
Storage Buffer Data Parity Bit: 551 TO 2230
Send Modem Initialization Text: NO
Delete Characters: NORMAL
Modem Port Set To: COM1

F1-Hlp F2-Msg F3-Bufr F4-Sav F5-Snd F6-Cfg F7-Clr F8-Print F9-Exit CM
```

ZENITH
data systems 
Groupe Bull

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

POSTMASTER: If undeliverable, please do not return.

\$2.50
P/N 885-2141