REMark®

Volume 8, Issue 12 • December 1987

P/N 885-2095 Issue 95

Challenging Games For The Z-100 (Page 85)

$2.50

EMULATE — Break the format barrier!

Read and write these disk formats:

Actrix	DEC Rainbow	Magnolia	Royal/Triumph
Altos	Eagle II	Morrow MD	Sanyo 1100
AMPRO	Epson QX-10	NCR DecMate5	Superbrain
Beehive Tpr	Fujitsu CP/M86	NEC PC-8001A	Televideo
CDR Systems	IBM CP/M86	Osborne 1	TRS80 CP/M
Cromemco	IMS 5000	Otrona	Xerox 820
DEC VT180	Kaypro II	PMC MicroMate	Zorba

H37 version includes formatting capability.

H89 or H8 with H37 ..... $59
For CDR BIOS 2.91 ..... $49

CPC CP/M ⇐ ⇒ PCDOS

Transfer files between PCDOS and CP/M disks. Includes access to subdirectories. For H89 or H8 with soft sector controller.

CPC — (H37, CDR or Magnolia) $35

ZCPR3 for H89 and H8

We are licensed by ECHELON to distribute the Z-System. Includes ZCPR3 system, ZRDOS + , supporting utilities, ZCPR3 manual by R Conn, ZRDOS manual and additional instructions. Comes installed with a bootable disk ready to run.

Z-System — specify format and hardware $98

4MHz mod for H89

Easy to install plug-in module. No trace cutting or soldering required. Includes CP/M software. ~~$45~~ **$39**

Assembled and tested — specify disk format ...
HDOS software ..... $ 5

6MHz mod for H89

Provides maximum high speed performance presently available for the H89 or H89A.

Assembled and tested — specify disk format ... $59

REP3 — Automatic Key Repeat

Modernize your H89 or H19. Hold any key down for half a second and the key begins repeating. Simple plug-in installation.

Kit $35
Assembled $45

Diskettes — Box of Ten:

5" DSDD hard sector $10
5" DSDD soft sector $10
8" SSDD (limited quantity) ..... $ 5

TIM2 — Real Time Clock

Installs in the left hand expansion slots of the H89. Includes battery backup. Requires soldering 4 wires to the CPU board.

Kit $65
Assembled and tested $75
Software (HDOS or CP/M) — specify format $10

DATESTAMPER

Product of Plu*Perfect Systems. Provides automatic time and datestamping for CP/M 2.2 files Limited Quantities! ~~$45~~ **$39**

For CP/M — specify disk format $45

Check or Money Order — Visa, M/C — C.O.D.
Add $4 per order for shipping and handling
California residents add 6% tax
Call or write for catalog, Monday through Friday, 9-5

ANAPRO

805/466-1589
6905 El Camino Real #4
Atascadero, CA 93422

FBE Products

For the H/Z-150, 160 Series

MegaRAM-150 — Modification kit allows memory board to be filled with 256K RAM chips (1.2 MByte). No soldering. Supplied with RAM disk software. ***$49.95***

ZP640 PLUS — Replacement PAL for standard memory board allows up to 2 banks of 256K and 2 or 3 banks of RAM chips to be installed for 640K or 704K maximum memory. ***$24.95***

COM3 — Replacement PAL allows installation of three serial ports (one an internal modem). Supplied with printer driver software for 3rd port. ***$39.95***

FBE Smartwatch

Calendar/Clock using Dallas Semiconductor's DS1216E SmartWatch module. Works with H/Z-110/120, 138/148, 150/158. Package includes SmartWatch with our software and documentation. Spacer kit ($2) required for Z-100. ***$44.95***

For the H/Z-100 Series

ZMF100a — Modification package allows installation of 256K RAM chips in older Z-100 without soldering. Works only with old-style motherboard. ***$65***

ZRAM-205 — Kit allows 256K RAM chips to be put on Z-205 memory board to make 256K memory plus 768K RAM disk. Requires soldering. PAL ($8) required for new motherboard. ***$49***

For the H/Z-89, 90 Series

SPOOLDISK 89 — 128K byte electronic disk and printer interface/spooler card. ***$195***

H89PIP — Dual port parallel interface card. Use as printer interface. Driver software included. ***$50*** Cable ***$24***

SLOT4 — Extender card adds 4th I/O expansion slot to right side bus. ***$47.50***

FBE

FBE Research Company, Inc.
P.O. Box 68234, Seattle, WA 98168
(206) 246-9815, M-F 9-5

UPS/APO/FPO Shipping Included.
VISA or MasterCard Accepted.

IBM-PC IN An H / Z-100

Basic Board W / HUG Software $149.00
HUG ZPC V2 & UPGRADE
Both Included
An $80.00 Value
This EXCELLENT COMBINATION is used by many universities, colleges and the U.S. Naval Academy.

Options Available
•No Solder H / Z-100 Mod Kit $5.
•COM 1 $44. •COM 2 $39. •Clock $44.
•8Mhz V20's $11.
•8Mhz Interrupt Kits, $12.

H / Z-100 Requires Modifications and 768K of RAM

CHECKS AND M.O. IN US FUNDS, VISA AND MC ACCEPTED
US ORDERS ADD $4.00 S & H, APO & FPO ADD $7.00 S & H
CA RESIDENTS ADD SALES TAX

IBM-PC is a resgistered trademark of IBM Corp.
ZPC is a product of the Heath Users Group

OTHER MFR'S PRODUCTS at LOWEST PRICES

HARD DISK DRIVES

Seagate
20mb ST225 ..... $259. • (w/cont.) $319. • (w SCSI cont.) ... $410.00
30mb ST238 ..... $279. • (w/cont.) $365. • ST4038 ..... $555.00
Z / 100 ST225N + CDR-11B Interface Board $725.00

MiniScribe
20mb M8425 ..... $275. • (w/SCSI cont.) $425. • M3425 ... $264.00
30mb M8438 ..... $299. • M3438 ..... $289.00
Z / 100 M8425S + CDR-11B Interface Board $745.00

Call or write for information on other sizes & brands.

CDR-ZS100 Speed Module $45.00
GRAYMATTER Application Software Z-MAX $49.00
FBE Research Company, Inc. ZMF100a $59.00

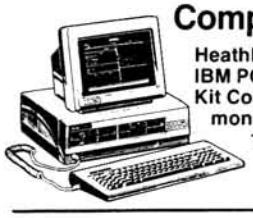
SPECIAL ANNIVERSARY OFFER
SL WABER: 6 Outlet Power Strip
15AMP • Surge & Noise Suppression $12.50

All prices are plus S & H
Call or write for further information

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

GIVE A HEATHKIT GIFT

IBM Compatible Computer Kit



Heathkit HS-148-H IBM PC Compatible Kit Computer with monitor and 1 drive

$499

SAVE 27-inch Stereo Color TV

Heathkit GRS-2700-2 Stereo TV Kit with Mediterranean Cabinet

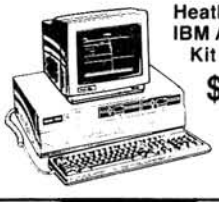
$499

Multimeter

Heathkit IM-2320 Heavy-duty Handheld Digital Multimeter Kit

$49

IBM AT Compatible



Heathkit HS-248-S IBM AT Compatible Kit Computer

$1599

SPECIAL PRICE Logic Analyzer

Heathkit IC-1001 Logic Analyzer Kit

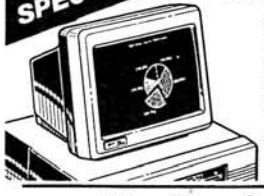
$199

Ham Gear

Heathkit HN-31-A Cantenna Dummy Load

$14.44

SPECIAL Monitor Kit



Heathkit HVM-1220-A 12" Diagonal Monochrome Monitor Kit

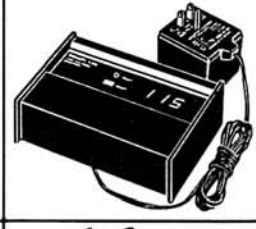
$40

Heathkit Digital Clock

Heathkit GC-1108 Digital Clock Kit

$19

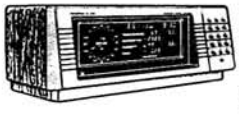
Line Monitor



Heathkit IM-2203 AC Line Monitor Kit

$39

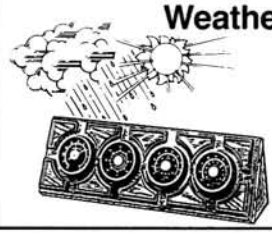
Weather Computer



NEW Heathkit ID-5001 Advanced Weather Computer Kit

$499

Weather Station



Heathkit ID-1290 5-Function Weather Station Kit

$149

Fish Caller

Heathkit SK-105 Fish Caller Kit

$4.95

Available NOW at Heath/Zenith Computers & Electronics Centers in the U.S.

PHOENIX, AZ
2727 W. Indian School Rd.
602-279-6247

TUCSON, AZ
5616 E. Broadway
602-745-0744

ANAHEIM, CA
330 E. Ball Rd.
714-776-9420

SAN JOSE (CAMPBELL), CA
2350 S. Bascom Ave.
408-377-8920

EL CERRITO, CA
6000 Potrero Ave.
415-236-8870

SAN DIEGO (LA MESA), CA
8363 Center Dr.
619-461-0110

LOS ANGELES, CA
2309 S. Flower St.
213-749-0261

POMONA, CA
1555 N. Orange Grove Ave.
714-623-3543

REDWOOD CITY, CA
2001 Middlefield Rd.
415-365-8155

SACRAMENTO, CA
1860 Fulton Ave.
916-486-1575

WOODLAND HILLS, CA
22504 Ventura Blvd.
818-883-0531

DENVER (WESTMINSTER), CO
8725 Sheridan Blvd.
303-429-2292

JACKSONVILLE, FL
9426 Arlington Expressway
904-725-4554

HIALEAH, FL
4705 W. 16th Ave.
305-823-2280

PLANTATION, FL
7173 W. Broward Blvd.
305-791-7300

TAMPA, FL
4019 W. Hillsborough Ave.
813-886-2541

ATLANTA, GA
5285 Roswell Rd.
404-252-4341

HONOLULU (PEARL CITY), HI
98-1254 Kaahumanu St.
808-487-0029

CHICAGO, IL
3466 W. Devon Ave.
312-583-3920

DOWNERS GROVE, IL
224 Ogden Ave.
312-852-1304

INDIANAPOLIS, IN
2112 E. 62nd St.
317-257-4321

KANSAS CITY (MISSION), KS
5960 Lamar Ave.
913-362-4486

LOUISVILLE, KY
12401 Shelbyville Rd.
502-245-7811

NEW ORLEANS (KENNER), LA
1900 Veterans Memorial Hwy.
504-467-6321

BALTIMORE, MD
1713 E. Joppa Rd.
301-661-4446

ROCKVILLE, MD
5542 Nicholson Lane
301-881-5420

PEABODY, MA
242 Andover St. (Rt. 114)
617-531-9330

WELLESLEY, MA
165 Worcester St. (Rt. 9)
617-237-1510

FARMINGTON HILLS, MI
29433 Orchard Lake Road
313-553-4171

EAST DETROIT, MI
18149 E. Eight Mile Rd.
313-772-0416

ST. JOSEPH, MI
2987 Lake Shore Drive
616-982-3215

MINNEAPOLIS (HOPKINS), MN
101 Shady Oak Rd.
612-938-6371

ST. PAUL, MN
1645 White Bear Ave.
612-778-1211

ST. LOUIS (BRIDGETON), MO
3794 McKelvey Rd.
314-291-1850

OMAHA, NE
2311 N. 90th St.
402-391-2071

OCEAN, NJ
1013 State Hwy. 35
201-775-1231

FAIR LAWN, NJ
35-07 Broadway (Rt. 4)
201-791-6935

AMHERST, NY
3476 Sheridan Dr.
716-835-3090

JERICHO, LI
15 Jericho Turnpike
516-334-8181

ROCHESTER, NY
937 Jefferson Rd.
716-424-2560

N. WHITE PLAINS, NY
7 Reservoir Rd.
914-761-7690

GREENSBORO, NC
4620C W. Market St.
919-299-5390

CINCINNATI (SPRINGDALE), OH
131 West Kemper Rd.
513-671-1115

CLEVELAND, OH
28100 Chagrin Blvd.
216-292-7553

COLUMBUS, OH
2500 Morse Rd.
614-475-7200

TOLEDO, OH
48 S. Byrne Rd.
419-537-1887

OKLAHOMA CITY, OK
7409 South Western
405-632-6418

PORTLAND (TIGARD), OR
10115 S.W. Nimbus Ave.
Nimbus Center
503-684-1074

FRAZER, PA
630 Lancaster Pike (Rt. 30)
215-647-5555

PHILADELPHIA, PA
6318 Roosevelt Blvd.
215-288-0180

PITTSBURGH, PA
3482 Wm. Penn Hwy.
412-824-3564

WARWICK, RI
558 Greenwich Ave.
401-738-5150

DALLAS, TX
12022C Garland Rd.
214-327-4835

FORT WORTH, TX
6825-A Green Oaks Rd.
817-737-8822

HOUSTON, TX
1704 W. Loop N.
713-869-5263

SAN ANTONIO, TX
7111 Blanco Rd.
512-341-8876

SALT LAKE CITY (MIDVALE), UT
58 East 7200 South
801-566-4626

ALEXANDRIA, VA
6201 Richmond Hwy.
703-765-5515

VIRGINIA BEACH, VA
1055 Independence Blvd.
804-460-0997

SEATTLE, WA
505 8th Ave. N.
206-682-2172

MILWAUKEE (WAUWATOSA), WI
845 N. Mayfair
414-453-1161

HUG discounts do not apply.

Prices good through December 24, 1987.

Units of Veritechnology Electronics Corporation

Authorized Zenith Data Systems and Hewlett Packard Dealer

Heath® ZENITH® Computers & Electronics

HZC-372

Heath/Zenith Users' Group

Managing Editor ..... Jim Buszkiewicz
(616) 982-3837

Software Engineer ..... Pat Swayne
(616) 982-3463

Software Coordinator ..... Nancy Strunk
(616) 982-3838

Production Coordinator ..... Lori Lerch
(616) 982-3794

Secretary ..... Margaret Bacon
(616) 982-3463

HUG Bulletin Board ..... (616) 982-3956

Contributing Editor .... William M. Adney

Contributing Editor ... Harold W. Bauman

Contributing Editor ..... Joseph Katz

Printer ..... Imperial Printing
St. Joseph, MI

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

* U.S. Funds

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

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

REMark®

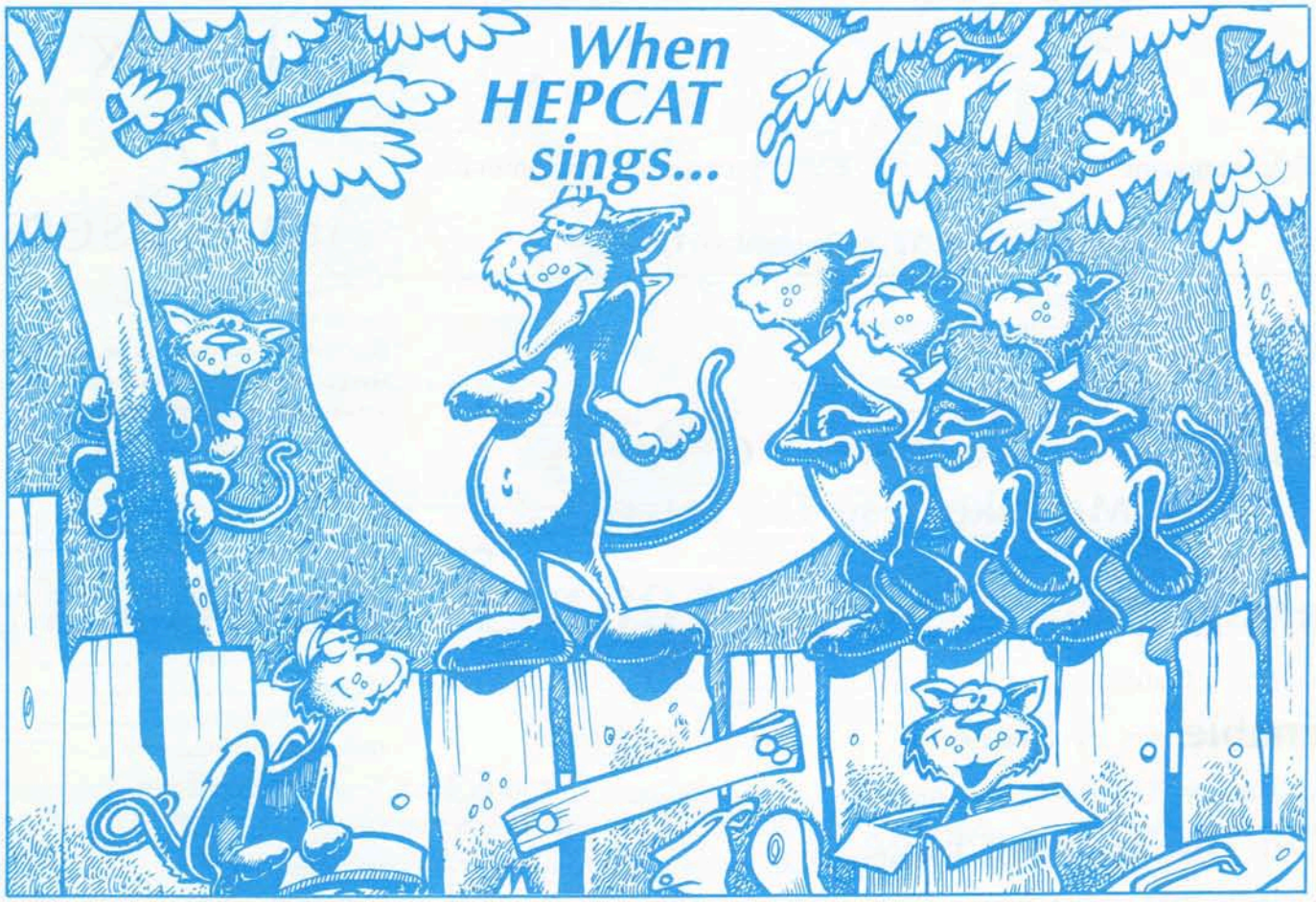
Official magazine for users of HEATH ZENITH computer equipment.

Volume 8, Issue 12 • December 1987

Index of Advertisers

This index is provided as an additional service. The publisher does not assume any liability for errors or omissions.

On The Cover: *Do you want to have fun this Holiday Season? Well, take a look at the fun and challenging games for your PC (Pages 9, 47, and 55). Along with these games, there are also a few for your Z-100 (Page 85). Photo by Jim Buszkiewicz.*



The other cats get to sing along!

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

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

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

HEPCAT gets along with everyone . . .

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

HEPCAT works harder . . .

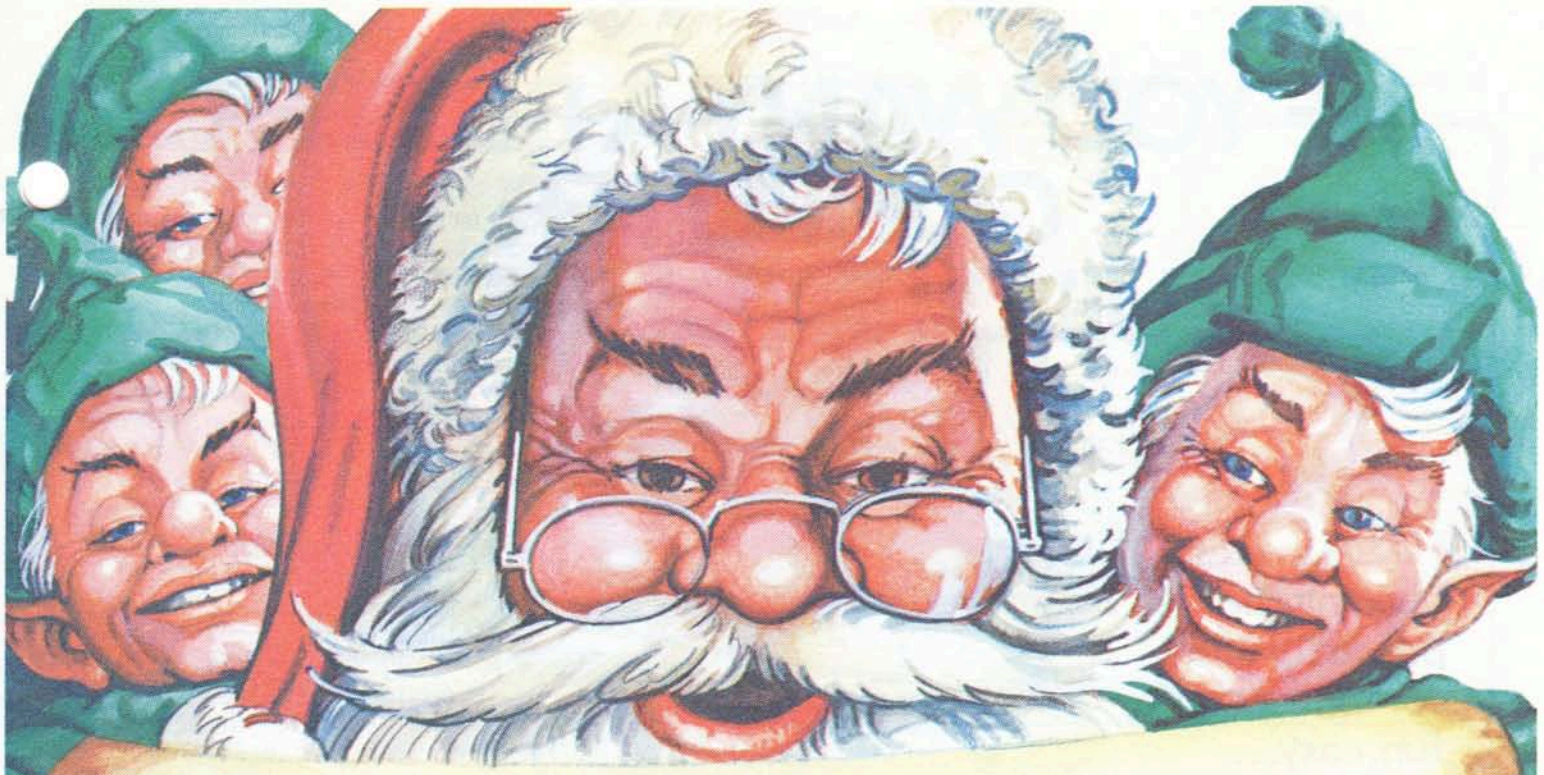
*HEPCAT provides a multi-function floating point calculator and a programmer's binary calculator that work together to do more than the basic four (+, -, *, /). The floating point calculator includes the following built-in functions: powers, pi, factorial, square root, sine, arc sine, cosine, arc cosine,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

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

HEPCAT eats less . . .

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

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



And The Winner Is...

If you remember correctly, we ran a little contest in the September issue of REMark. The contest was easy. Guess how many black disks were pictured on the front cover. It's amazing how close many of you came to the actual count of 99! But, as they say, close only counts in horseshoes and handgrenades. The person who won all those disks was James G. Wagner, with his guess of exactly 99. Second prize, a copy of Pat Swayne's *HEPCAT* program, went to H. Dopheide, with a guess of 98. The third place winner was Ralph Henrichs, and he will receive a copy of *HADES*. His guess was 100. Honorable mention goes out to Charlie Dischinger for the most amount of entries sent in . . . all of them wrong, of course. Congratulations to those winners, and a hearty thank you to those of you that participated.

Our bulletin board service, *HUGPBBS*, is getting quite exciting now that we've added a Bargain Centre. More items are being added (and sold) every day, at un-heard of prices. If you're a HUG member, it costs nothing to become registered. Just give the board a call at (616) 982-3956 (with your modem), or give me a call at (616) 982-3837 (with your mouth). Many of you have tried to call me personally with your modems. Believe me, this will not work; my ears sharply cut off at 1100 Hurtz!

Everyone, young or old, at one point in his life, plays a game of some sort. Computer users are no exception. In this issue, I decided to review a few of the hundreds of games which are available for the H/Z computer systems. In fact, you might even call this our Gamey Christmas issue.

Yes, our reader service is in place, and working! Keep those cards coming!

If you received a plastic ID card, and a paper ID card, chances are good that the ID numbers are NOT identical. Discard your plastic card. Those cards are remnants of the old Heath database system.

From all of us here at HUG, we'd like to thank you for your continued support, and to wish you and your family the very happiest of this holiday season.

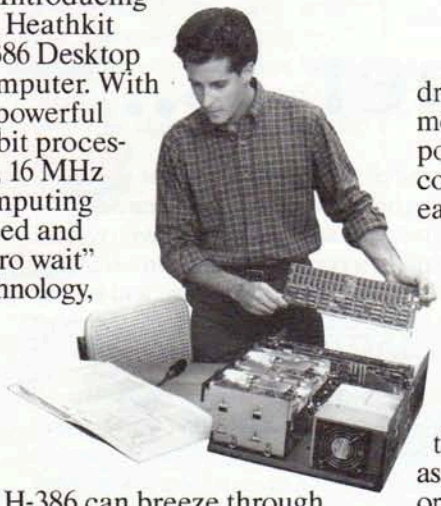
Jim Buszkiewicz
HUG Managing Editor

IF YOU WANT A BETTER COMPUTER BUILD IT YOURSELF.

Heath makes building the powerful 386 computer quick, fun, easy... and a great value.

Now you can own the fastest, most powerful home computer available today - at an affordable cost. Just by building it yourself.

Introducing the Heathkit H-386 Desktop Computer. With its powerful 32-bit processor, 16 MHz computing speed and "zero wait" technology,



the H-386 can breeze through complex calculations in seconds.

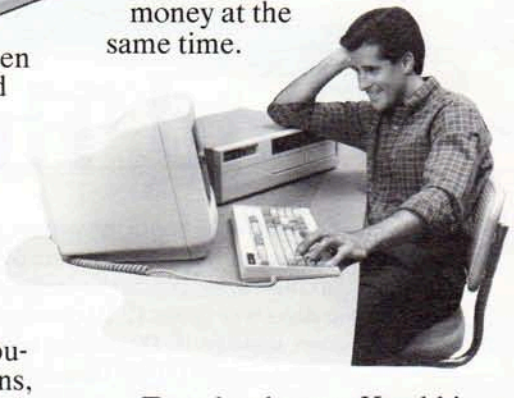
Every INTEL 80386 microprocessor used in our H-386 is 100% tested for all functions. And you get superb graphics because one video port provides dazzling 640-by-480 color on Zenith's new flat screen monitor and another port drives EGA, CGA and TTL monochrome monitors. Both ports automatically emulate common video formats for easy system configuration.

Designed for people using large spreadsheets, CAD/CAM or other computation-intensive applications, or anyone who simply wants to own the newest, most powerful hardware on the market, the Heathkit H-386 can be assembled easily and quickly. One or two evenings is all it takes, and no special tools or equipment are required.

In the bargain, you get the satisfaction of having built a powerful computer system all by yourself, and the confidence that this exciting product will deliver all the performance and dependability you expect. At a significant savings over comparable off-the-shelf brands.

What's more, all Heathkit products are backed by a newly extended, limited one-year warranty, highly respected manuals and technical consultation service.

So if you want a better computer, build it yourself. Impress your friends. And save money at the same time.



To order the new Heathkit H-386 Desktop Computer, simply call toll-free **1-800-253-0570. Ask for operator 611.** Use your Visa, MasterCard or Heath Revolving Charge. Or call **616-982-3614** for the Heath/Zenith Computers and Electronics store location nearest you.

For more information on all our quality kits, send now for your free four-color Heathkit catalog. **Write Heath Company, Dept. 016-594, Benton Harbor, MI 49022.**

Prices, product availability and specifications are subject to change without notice.

Heathkit®

Heath Company

Heathkit is a registered trademark of Heath Company, a subsidiary of Zenith Electronics Corporation.

War Simulations for Adults

by Joseph Katz

Although I'm usually not much interested in computer games, I find MicroProse products for mainstream computers curiously attractive. These aren't kid stuff or arcade games: they're war simulations for mature adults. Even though I'm not bellicose these simulations fascinate me because they're rooted in historical reality. I command situations that start from the elements of an important battle. Its outcome most often depends on my decisions more than my reflexes. There's action, of course, but in most cases my reflexes don't have to be those of a younger man. Thought and planning are more important.

Silent Service, for example, commands a submarine in the Pacific during World War II. You choose among six different convoy scenarios and five different patrol scenarios. Convoys are easier than patrols because they're shorter and the targets are determined by history: you're Lt. Commander White of the *Plunger*, Lt. Commander Morton of the *Wahoo*, Commander Martin of the *Hammerhead*, Commander Cassey of the *Searaven*, Lt. Commander Sieglaff of the *Tautog*, or Lt. Commander Moore of the *Grayback*. You can vary circumstances such as whether some of your torpedos are duds or whether the enemy can zig zag or fight back. You can opt to do target practice, and you're sent back to the practice range if you can't identify the silhouette of a specific enemy vessel that is your passage to sea duty when you opt to go on patrol. (It's really a way MicroProse wards off software pirates.)

Conflict in Vietnam gives you control of crucial battles in that sad war: Dien Bien Phu (1954), Ia Drang (1965), Khe Sanh (1968), Cambodia (1970), Quang Tri (1972). Variants of those five battles let you explore alternatives, including those that would have faced you on the other side. It gives me the same odd feeling I recall from *The Wheels of If* by, I think, L. Sprague DeCamp many years ago. What, for example, if the French had had American air support at Dien Bien Phu? (I won.) You command simulated real troops in simulated real time, the action depending on conditions such as weather, readiness, supplies, and fatigue, from a broad perspective looking down on the entire combat area. But I find it impossible to maintain complete intellectual objectivity about what's happening. The casualties are frightening. It's reality.

F-15 Strike Eagle is too, in quite another way. There are seven combat missions in places, such as the Persian Gulf, that are in the news today. It's depressing to learn that I couldn't possibly pilot an F-15 jet. The machine is too fast for me. I crash a lot.

Gunship, the PC version of a bestselling airborne simulation for other computers, is more my speed. It's also a knockout on a system with an EGA display and hard disk. Unlike the other simulations, that can only be run from their copy protected floppy disks, you can install Gunship's two disks on one hard disk. (You can deinstall it if necessary.) Your gunship is an attack helicopter armed with current weapons and is astonishingly realistic to operate. Not only the machinery but also the scenery seem real: MicroProse is justifiably proud of the smooth, three-dimensional effects in this simulation. There are specific missions to accomplish, with briefings and maps to guide you, and the outcome depends on your skill and judgment. If they're good enough, you attain rank. Otherwise you are

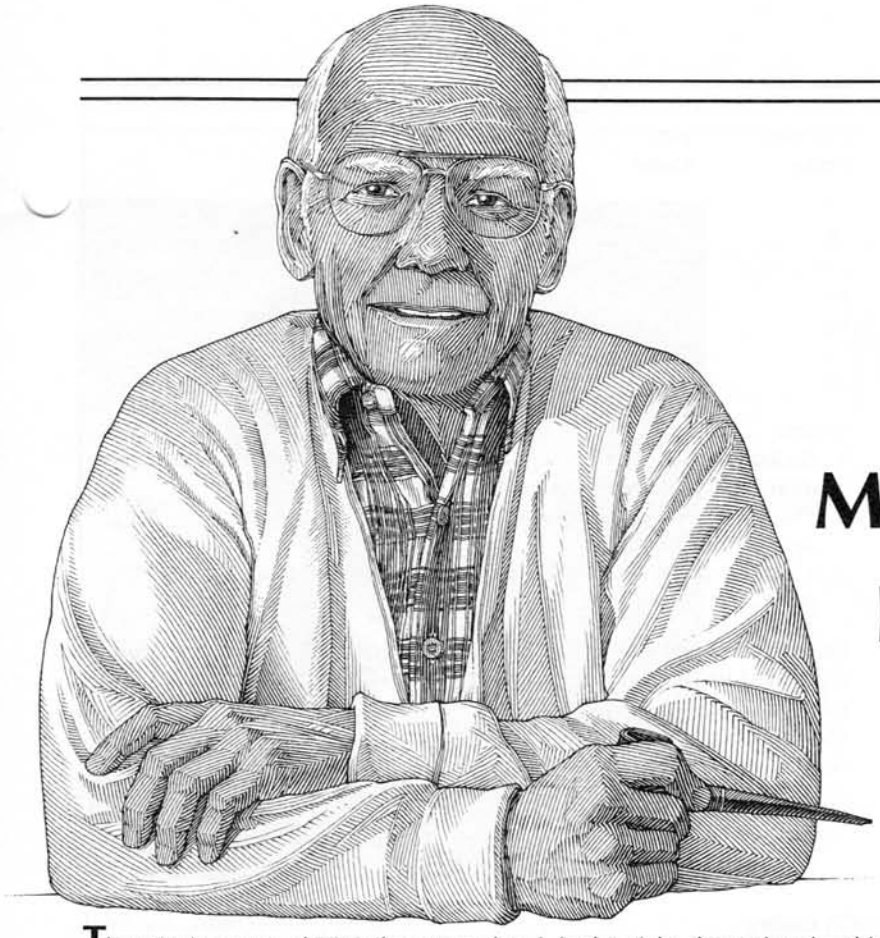
captured or die. On one occasion or another I've been promoted, taken, or buried.

Those are serious outcomes seriously presented. Because the consequences of war are not trivialized, I don't have to worry about nine-year-old Matthew being dehumanized. That's what I do worry about when he's exposed to other war games. MicroProse battles, like the real thing, are dangerous and potentially deadly. They are serious business. You therefore must be prepared to spend time studying the manuals. The manual for Silent Service, as an example, is a fascinating introduction to submarines during World War II, duty on them during phases of the Pacific war, and tactics used then. All these MicroProse simulations take learning and increase it.

And these simulations run beautifully on Zenith equipment. I've used all on a Z-248, and all but Gunship so far on a Z-171. (They show up surprisingly well on the laptop's liquid crystal display screen, which makes the combination a winning travel companion.) You'll need either a CGA (standard on Zenith's PC-100 series) or EGA display (Gunship is dazzling on EGA). The simulations all take advantage of a joystick, if you own one, but can be used effectively from the keyboard too.

Here's more fun than a spreadsheet.

Conflict in Vietnam. $39.95.
F-15 Strike Eagle. $34.95.
Gunship. $49.95.
Silent Service. $39.95.
MicroProse
120 Lakefront Drive
Hunt Valley, MD 21030
301/771-1151 *



ASCII

We See It Mentioned Often, But What Is It?

Kenneth Mortimer, PE
352 Green Acres Drive
Valparaiso, IN 46383

There isn't a copy of REMark or any other respectable computer journal in which the acronym ASCII does not appear. The acronym also appears in the body of many computer textbooks, but other than a usually undecipherable table in the appendix of the text, there is very little explanation of the term. It is something the experts think is so obvious that we all know everything about it.

What is this AS KEY that computer people talk about so often? Where does this "CHR$(27)" in BASIC programs come from? What does all of this mean to each of us who sit in front of a terminal and where does it come from?

In the early part of the 1960's, the codes for the transmission of information were many in number and confusing in nature. Perhaps the most common was the BAUDOT code used in the teletype machines. While well established, it had the disadvantage in that the same code could represent either a character or a figure, depending upon whether the machine was in "letters" or "figures" mode. The code 10100 represented the letter "U" in letters mode and rang the warning bell in the "figures" mode. Since a teletypewriter had no provision for lower case type, Baudot had no provision for lower case characters. The Baudot code was not only used for the punched paper tape, but also for transmission. The five digit Baudot code had a strong competitor in the eight channel code which was used in both the Freiden Flexewriter and the IBM 046 tape punch. The unit record card also had three major competitors: the 80 column by 12 row Hollerith (IBM) code, the 90 column by 6 row Univac card and the British Bull code.

It was recognized that a standard code would soon be required for the communication between the components of a system, between systems, and even internally within a component of a system. On June 17, 1963 the first version of the American Standard Code for Information Interchange (hence ASCII) was adopted by the American Standards Association X3.2 Subcommittee. It was a seven bit code with some limitations and contained many codes required for telecommunications, including such codes that are now anacronistic to those in the computer area, but were essential twenty years ago. If you will look at the table of ASCII characters in your Heath/Zenith computer manual, you will find such codes as "RU" (are you ....?) or "SYNC" (synchronous idle) which show their teletype heritage. The H/Z manuals are among the few places that both the "nonprinting" and the "printing" codes are shown with a description of the "nonprinting" codes.

It was soon recognized that the seven bit code (having upper case letters only) was severely limited, so an extension of the ASCII code to 8 bits was adopted in 1967.

Since the ASCII code is organized on the basis of bit pattern, it is most easily explained in terms of the hexidecimal number system. However, we are accustomed to thinking in the decimal number system, and since the BASIC and Pascal functions that access the ASCII codes use decimal numbers, I will use both number systems, but will follow the decimal number with "D" and the hexidecimal number with "H".

The Nonprinting Codes

The first 32D (0D/0H to 31D/1FH) ASCII characters are nonprinting characters. When they are used, most of them normally cause no output in the monitor. I will show you later how to test the output of the ASCII codes with a simple BASIC program. I have tried it with two combinations; A Z-89 with a Daisywriter and both a Z-148 and a Z-150 with a Panasonic KX P1092 printer. I will discuss the results as we go along. Normally, one would expect most of these characters to have no effect on the CRT screen. However, some ingenious computer designers have found that they can store some fancy graphic characters in the character generator that can be addressed by the "nonprinting" ASCII codes not needed for their primary purposes. The Z-150 has some neat ones hidden in the

character generator, but they will not be transmitted to the printer. Some of the codes that are common to most computers, and will be transmitted to the printer, are shown in Table 1.

The Printing Codes

The tables in the back of most computer texts contain a table of the printing ASCII code. Each author tries his or her best to generate a table that will be of the greatest use to them. My own version of the table of printing codes is shown as Table 2. Since the code is based on a bit pattern, it is most easily understood when each column represents a hexidecimal number with the same highest order digit. I previously mentioned that the first 32D numbers represented "nonprinting" characters. These would be hexidecimal numbers with 0 and 1 as the high order (or left hand) digit. Since Table 2 consists of only the printing characters, the columns which would have 0 and 1 in the high order position when listed in hexidecimal form do not appear since they contain the nonprinting characters. Now let us study Table 2. The first column contains special symbols. The second column contains decimal numbers plus more special symbols. Note that the low order (right hand) digit of the code in hexidecimal form is the decimal value of the number (30H is the number 0 and 39H is 9). Also, note that very often the two characters that occupy the same key are in the same row in two left-hand columns. The explanation point (!) is 21H and the digit 1 is 31H. The greater than symbol (>) is 3EH and the period is 2EH. The designers of the code attempted to have only one bit different between characters that would normally occupy the same key on a typewriter keyboard.

When we examine the third and fourth column (those whose high order hexidecimal digits are 4 and 5), you find that they contain the upper case letters used in the English (Roman) alphabet in alphabetical order. Since there are only twenty-six letters in our alphabet, there are six codes left which are used for some more of the symbols. However, if you speak a language "mit umlaut" or "con tilde", there is a good chance that codes 5BH to 5FH will include the upper case letters used in your language that are not in the twenty-six used in the English-speaking world. Note that the two right-hand columns contain the lower case letters. Again, the same comment will apply to these that applied to the middle two. The ASCII code for a lower case letter is therefore always 20H or 32D greater than its upper case equivalent.

Decimal Code	Hex Code	Description
7	7	Rings the "bell" on terminal only
8	8	Back Space (1)
9	9	Horizontal tab
10	A	Line feed
12	C	Form Feed (2)
13	D	Carriage Return (3)
27	1B	Escape

Notes:

1. Backspace will erase the preceding character on the monitor. It will cause an overstrike on the printer.
2. The form feed will probably cause no action on your monitor.
3. 13D/DH caused both a carriage return and a line feed on the Z-150.

Be very careful using the code 19D/13H because the XOFF command will turn off communication between the computer and the printer. Depending on the method of "handshaking" between the computer and printer, you could "crash" your system.

Table 1
Selected "Nonprinting" Codes

Using The ASCII Codes

Let's take a look at a few BASIC programs that will permit us to examine some of the many uses of the ASCII codes. Here is the coding for the program in CP/M or GW versions of MBASIC:

```
100 REM A PROGRAM TO PRINT THE
    EQUIVALENT OF THE ASCII CODES
110 PRINT "ENTER A NUMBER BETWEEN
    0 AND 127"
120 INPUT N
130 IF N > 255 THEN END
140 PRINT N,"(";CHR$(N);")"
150 LPRINT N,"(";CHR$(N);")"
160 GOTO 110
```

If you are using BHBASIC or the HDOS version of MBASIC, you will have to modify Line 150 and add the appropriate opening statements to permit outputting to the printer.

Lines 140 and 150 are the key to the program. The first item in the output string is an echo of the input variable so that both the monitor and the printer will show the decimal value of the ASCII code. The comma following the N causes the cursor or print head to skip to the beginning of the next print field. The next item in the output string is an opening parenthesis "(" as a literal. It is followed by a semicolon so that the cursor or print head does not skip to the next print field. The third item in the output string is the BASIC function that converts the value N to its ASCII equivalent. It is again followed by a semicolon. The final item in the output string is the closing parenthesis as a literal. The result is a line containing the numeric value of the particular ASCII code, followed by its equivalent enclosed in parentheses. If there is no conversion of the number to its ASCII equivalent, the two parentheses will be adjacent. If the backspace occurs, the monitor will show only the closing parenthesis, while the printer will show an overstrike of the two parentheses. Tabs, line feeds and carriage returns will be obvious from the position of the opening and closing parentheses.

AGAIN, I MUST REPEAT THE WARNING ABOUT USING THE DECIMAL VALUE NINETEEN. The XOFF command does weird things to the communication between the computer and the printer. You may end up having to turn off both the computer and the printer to recover from the problem that this transmission causes. Let us enter the program and see what happens.

The Output Of The Nonprinting Codes To The Monitor Of The H-89

When the first 32 decimal values are entered, most of them cause no output to the screen of the '89. The two parentheses just follow one another. The following outputs are noted:

```
 N    Code            Screen Indication
 7    Bell            rings the bell
 8    Back Space      )       [the opening parenthesis is overstruck]
 9    Horizontal tab  (       ) [the second parenthesis tabbed over]
10    Line Feed       (
)                                [line feed and carriage return]
12    Form Feed       skips 13 rows but does not cause return
27    Escape          (       [I can not find out what "Esc )" is but
                               it does not print or show any change
                               in the next line]
```

Note that ASCII 27 generates the escape code and the result is the same as pressing the ESCAPE key. That is why you see CHR$(27) in so many BASIC programs.

The Output Of The Nonprinting Codes To The Monitor Of The Z-150 & Z-148

It seems that the designer of the character generator in the Z-150 has decided to store a whole bunch of neat graphic characters in that ROM. When you use the nonprinting ASCII codes, they are most intriguing. Among (and only some) them are:

```
 2    a face
 3    a heart
 4    a diamond
 5    a chess king
 6    a chess pawn
24    an up arrow
25    a down arrow
and many others.
```

It should be noted that these symbols are not transmitted to the printer. The same character generator is used in the Z-148.

The Nonprinting Codes And The Printer

The nonprinting codes can be useful in controlling the printer. Eight, the backspace code, causes the printer to backspace and then print the next character. This will permit you to overstrike or underline a character. Nine, the horizontal tab code, causes the print head to tab over to the next set tab location.

Ten will either generate a line feed or a line feed and a carriage return. Eleven will generate a vertical tab on some printers. Twelve generates a form feed and advances the paper to the top of the next page. Thirteen will generate a carriage return, but no line feed on some printers. Fourteen through eighteen are used by some dot matrix printers to vary the pitch (horizontal width) of the characters.

WATCH OUT FOR THE EFFECT OF NINETEEN!!!!!!!!!!!!! XOFF can cause trouble. Since printers and their device drivers can be configured to respond to these nonprinting codes in various ways, only a check using the test program or thorough reading of the instruction manual will tell you exactly how your particular combination of printer and device driver will respond to a particular code. Since the dot matrix printers have their own character generating ROMs, do not expect the characters that the non-printing codes place on the monitor screen to be copied by the printer, even when you depress the screen copy key.

The Printing Codes And The Monitor

If the character generator in your monitor is designed to generate the standard English language characters, you should have no problem and the response should match the Table. If you have a monitor with a character generator configured for some other western language, some of the special characters will be replaced by the appropriate letters. The MS-DOS KEYBXXXX command permits you to replace the character generating ROM with a set character generator for several languages that can be stored in RAM. These commands convert the keyboard of the computer to a keyboard equivalent to the typewriter keyboard for the particular language.

The Printing Codes And The Printer

Normally, one would expect the printer to respond to the printing codes in a one to one match. There are some problems that can occur. If the printer is configured for some language other that English, some of the special characters will be replaced by characters from that language. Some impact printers come with print wheels or thimbles that do not respond to the ASCII codes, but were designed for typewriter applications. These will print the characters as they would appear on a normal typewriter keyboard and not on the computer keyboard. One of the print wheels for my Daisywriter is set up in this manner and among other things, prints a superscript 2 in place of the karat "^". It makes program listing unreadable. However, most printers will respond to the printing codes in the expected manner. The MS-DOS KEYBXXXX command only changes the character generator for the monitor and not for the printer.

The Extended Character Set

Up to this point, we have only used 128 ASCII codes. The conversion to the eight bit ASCII coding system presents the opportunity for 256 different codes (0D to 255D). That is why line 130 in the BASIC program sets an upper limit of 255D. Integers greater than 255D will generate an error. These additional 128 codes may be used in different ways. The H-89, for reasons that I cannot phathom, repeats the first 128 ASCII codes. The Z-148 and Z-150 have a true extended character set in their character generator. These include many non-English (I should say non-Roman) letters, graphic symbols and such goodies as plus/minus (241), equal or greater than (242), equal or less than (243), the common division symbol (246) and the square root symbol (251). While these characters appear on the monitor screen, they are not printed by the Panasonic printer. Lotus 123

DEC	HEX	CODE	DEC	HEX	CODE	DEC	HEX	CODE	DEC	HEX	CODE	DEC	HEX	CODE	DEC	HEX	CODE
32	20	SP	48	30	0	64	40	@	80	50	P	96	60	`	112	70	p
33	21	!	49	31	1	65	41	A	81	51	Q	97	61	a	113	71	q
34	22	"	50	32	2	66	42	B	82	52	R	98	62	b	114	72	r
35	23	#	51	33	3	67	43	C	83	53	S	99	63	c	115	73	s
36	24	$	52	34	4	68	44	D	84	54	T	100	64	d	116	74	t
37	25	%	53	35	5	69	45	E	85	55	U	101	65	e	117	75	u
38	26	&	54	36	6	70	46	F	86	56	V	102	66	f	118	76	v
39	27	'	55	37	7	71	47	G	87	57	W	103	67	g	119	77	w
40	28	(	56	38	8	72	48	H	88	58	X	104	68	h	120	78	x
41	29	)	57	39	9	73	49	I	89	59	Y	105	69	i	121	79	y
42	2A	*	58	3A	:	74	4A	J	90	5A	Z	106	6A	j	122	7A	z
43	2B	+	59	3B	;	75	4B	K	91	5B	[	107	6B	k	123	7B	{
44	2C	,	60	3C	<	76	4C	L	92	5C	\	108	6C	l	124	7C	|
45	2D	-	61	3D	=	77	4D	M	93	5D	]	109	6D	m	125	7D	}
46	2E	.	62	3E	>	78	4E	N	94	5E	^	110	6E	n	126	7E	~
47	2F	/	63	3F	?	79	4F	O	95	5F	_	111	6F	o	127	7F	DEL*

* DEL is not used in most computers

Table 2
The Printing Codes

has its own extended character set. If your operating system has the GRAPHIC and GRAFTBL function, you may be able to print these characters on a dot matrix printer.

How To Access The ASCII Codes

BASIC, Pascal and Lotus have functions that allow one to convert the numeric value of the code to its ASCII equivalent and the ASCII equivalent to its numeric value. In BASIC, they are the ASC function for converting a symbol to its numeric equivalent and the CHR$ function which will convert the numerical value to its equivalent symbol. The ASC function is part of the ASCII standard BASIC and should appear in every dialect of BASIC. While its inverse, the CHR$ function is seen more often in programs. It is not part of the standard BASIC and might have other forms in other dialects of BASIC. In Pascal, the corresponding functions are ORD, for ordinal value, and CHR. These functions only correspond directly when the internal representation of the characters is in the ASCII code. Lotus uses the CODE and the CHAR function for the same purposes.

For What Purposes Do I Use These ASCII Codes?

There are many uses for the conversion to and from ASCII code.I will only show three simple examples in BASIC. If you prefer another language, you can easily convert to the language of your choice. I am certain that these will help you to find more uses on your own.

1. The generation of the ESCAPE code: The use of the CHR$ subroutine to generate an escape code in a BASIC program is commonly used. Assigning CHR$(27) to a string variable permits the use of the ESCAPE codes for screen or printer control. There have been so many articles in REMark about the use of the ESCAPE codes that I would show my prejudice if I were to list only a few.

2. To determine if a character is a number or a letter: Since the decimal numbers are represented by the ASCII codes 48D to 57D we can ask

```
IF ASC(n)>47 AND ASC(n)<58 THEN
```

   If the above statement is true, "n" is a decimal digit.

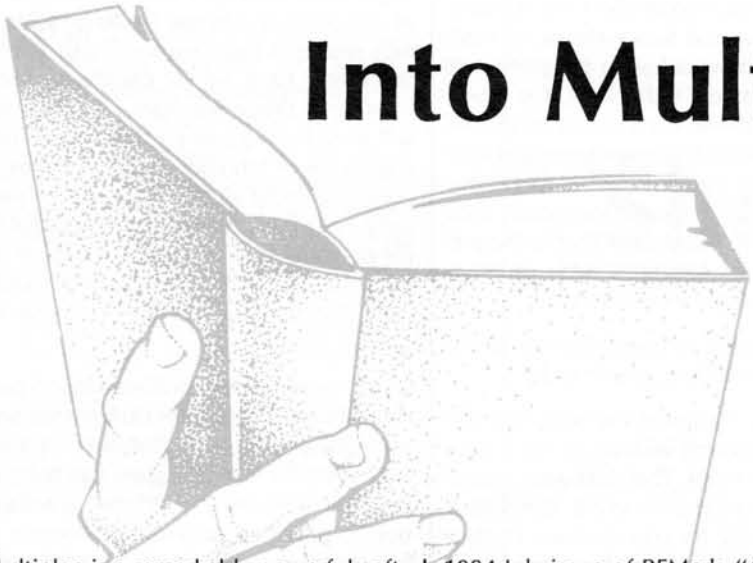
3. Convert a lower case letter to an upper case letter: Since the lower case letters are represented by the codes 97D to 122D and the upper case letters are represented by the codes 65D to 90D, one can convert a lower case letter to its upper case letter by subtracting 32D from the lower case code.

```
IF ASC(str) > 96 AND ASC(str) < 123
THEN str = CHR$(ASC(str) -32)
```

   where "str" is the letter to be converted to upper case if it is lower case.

I hope that this article leads you to a better understanding of what the ASCII codes are and what they can be used for. If you are not of the programming type, they will help you to increase your computer literacy. *

How To Read ASCII Files Into Multiplan



Robert E. Heath
9 East Dunnrobin Bay
Sault Ste Marie, Ontario
CANADA P6C 5T4

Multiplan is a remarkably powerful software tool with an abundance of features that can be used to accomplish a wide variety of tasks involving numerical data. Among Multiplan's features are the ability to both write and read ASCII files. The former is easily accomplished by means of the Print File command. The reading of ASCII files is quite another matter; because Multiplan needs to know what parts of an ASCII file are to be read into its cells, an extensive conversion process is needed. This article is intended to shed light on that process and make it as easy as possible for the reader.

First, a little bit of history. For some time, I've wanted to enter a lengthy ASCII inventory listing into Multiplan without having to enter it directly character-by-character and line-by-line. It was apparent that the job could be done by using the SYLK (Symbolic Link) format outlined in Appendix A3 of the Heath/Zenith Multiplan manual. However, I found that I just couldn't understand how to apply the information found there. Further, it appeared that the conversion of ASCII data to the SYLK format would require more time than simply entering the data into Multiplan directly. In retrospect, I can say that Appendix A3 of the Multiplan manual is an excellent reference once one understands how the SYLK process works, but is of questionable value as a means to show the way.

The first breakthrough towards the solution of my problem was the article in the 1984 July issue of REMark, "A Program to Transfer ASCII Files into Multiplan", by Ken Simmons. I eagerly applied Simmons' program to my ASCII file of inventory data only to find that the data as it appeared in Multiplan was quite chaotic and quite unusable. Nothing that I initially tried solved the problem and I eventually retreated from the project for many, many months before returning to examine the problem more closely.

As it turned out, the format of my input data was quite different than the example provided by Simmons in his article. It became apparent, therefore, that I would have to modify his program extensively to match the characteristics of my ASCII inventory file. By providing the listing of the program used for my particular application, along with two examples of how to modify it to fit two different data formats, plus some guidelines, I hope to show the way to write a program that will fit the format of any ASCII data that one might want to read into Multiplan.

Figure 1 is a listing of five sample lines of the format of the ASCII inventory file which I wanted to load into Multiplan. Figure 2 is a listing of the program that I used to convert that listing into a SYLK file that could be read by Multiplan. Figure 3 is a listing of the SYLK file produced by the program ASCII2MP.BAS using the inventory data of Figure 1 as input. Figure 4 is the inventory data as printed from Multiplan using the Print File command. The description I used for the six fields of the inventory listing were: Item Description, Manufacturer, Part Number, Unit (of measure), Quantity, and Unit Cost. Within Multiplan itself, I wanted a seventh field (i.e., a seventh column), Total Cost, which was the product of the fifth and sixth fields, i.e., Total Cost = Quantity x Unit Cost. That field is provided by program lines 770 and 900.

While there are a few readers that might have ASCII files that perfectly fit either Simmons' program or mine, the majority of readers most likely have ASCII files with different formats that will require program modifications in order to work properly. If modifications are necessary, it will likely be necessary to absorb at least some of the detail found in Appendix A3 of the Multiplan manual. However, in most cases, it should not be necessary to fully digest and understand everything found there in order to modify my program to adapt to most input ASCII file formats.

As any Science or Mathematics teacher knows, as well as any (modestly intelligent) student, it is far easier to analyze someone else's solution to a problem than it is to actually solve the problem without help. In this situation, Multiplan itself is able to provide the essence of the solution to the problem. Specifically, it is far easier to analyze an existing SYLK file in a desired format than it is to create one from scratch. Hence, the most valuable guidance I can give to anyone who has an input ASCII file which differs in its fields from those of mine follows.

Enter Multiplan and set up a file with 2 or 3 lines in the form that you would like to see your ASCII file appear in Multiplan. Forget about headings or any other fluff. (More about that later.) When your short sample Multiplan file is complete, use the Transfer Options Symbolic command followed by the Transfer Save command (with an appropriate filename) to create a SYLK file. Then, read that file into any word processor that reads ASCII files (most can) and examine the SYLK file. It will likely be convenient to print that file from the word processor to make examination easier. (Somewhat to my surprise, it seems that Multiplan cannot itself print the SYLK file.) The structure of the SYLK file will provide you with invaluable guidance towards the way in which ASCII2MP.BAS needs to be modified in order to create a SYLK file to fit the needs of your particular application.

Further to the previous paragraph, there is more than one way that a SYLK file can be written to accomplish any given task. My program does not create the most efficient SYLK file. Rather, my BASIC program has been written to create a SYLK file in a way that I believe is the easiest to program in BASIC and the easiest to understand. A sample SYLK file, using my data format and created by Multiplan, would differ somewhat in its format from the SYLK file created by my program, ASCII2MP.BAS. Yet, both would produce the same final result.

The content of the previous paragraph invites an explanation. When writing any computer program, irrespective of the programming language, there is always a tradeoff between efficiency on the one hand, and program simplicity and program writing time on the other. If a program is likely to be used hundreds or thousands of times, as the program for an operating system, efficiency is paramount and simplicity and ease of understanding is very much a secondary matter. On the other hand, a program that may be used only once, or only a few times, is far better written with simplicity and ease of understanding as a first consideration with efficiency given only secondary consideration. The program I have written emphasizes simplicity and ease of understanding because I believe that approach best suits the needs of most readers. Those who require efficiency should be able to write a program that meets such a need.

It would be possible to write a BASIC program that would incorporate all wanted headers, final totals, and whatever else, into the Multiplan SYLK file. Only if many essentially identical files were to be read into Multiplan would such programming details be justified. Otherwise, it is much easier to add the headers and other details after the SYLK file has been imported into Multiplan. The essential task is to get the many lines of identically formatted data transferred into Multiplan. That is the job done by the ASCII2MP.BAS program whose listing appears in Figure 2.

Some comments and explanations about the program itself are now in order.

The program masthead explains that the program works only with input ASCII data in lines with 6 fields. That data was created with a word processor using standard 8 column tabbing for convenience of data entry. Hence, all fields are 8 columns wide or multiples of 8 columns. An exception is columns 4 and 5 which together add up to 8 columns.

No commas, semicolons or tabs can exist in the source ASCII file. A comma is seen as a record terminator by BASIC and will, therefore, cause any line of input data to be broken up into two or more lines with resulting chaos in the Multiplan file. Many lines of my inventory data had commas in the "Item Description" field. To overcome that problem, I used my word processor to replace all commas with colons, quickly done by using a global replace command. Afterwards, I brought the final file, written by Multiplan, back into my word processor and replaced all the colons with commas. A semicolon cannot appear in the source file because the semicolon is used as a delimiter by Multiplan in SYLK files. As a matter of fact, if such a semicolon does exist anywhere in the source file, Multiplan will abort the reading of the SYLK file with the error message "File format error:". The existence of any tab character in the source file shortens the field lengths and once again will cause chaos in the resulting Multiplan file. You will recall that I originally created my ASCII source file by making extensive use of tabs to simplify data entry; hence, my original source file contained an abundance of tab characters. If your word processor is WatchWord, as is mine, that problem is easily and quickly cured by the single command BLANKIT* which causes all tab characters in the file to be filled in with spaces. I suppose some other word processors have similar capability, but if your word processor does not, then you will have to find some other way to rid a source file of tab characters. One way, of course, is to fill all tabs with spaces manually, not a particularly appealing task if the file contains large numbers of tabs. If you have the MS-DOS 3 Programmer's Utility Pack (PUP), the job can be done with the TABSPC Utility.

BASIC requires considerable time to process a lengthy input file. During that time, one can become anxious about progress or even wonder if the program has hung up. Program lines 440 and 490 set up a condition to provide status information on the screen in an appropriate location, on a single line and without scrolling. The actual status is provided by line 500. A similar function is performed by lines 790, 810 and 820 during the formation of the output SYLK file.

The array dimensions of line 180 and the upper index for the loop in line 460 are based on the fact that version 1 of Multiplan can handle a maximum of 255 rows.

Lines 510 through 560 break up each input record (=line in this application) into the 6 fields to prepare them for entry into the SYLK file. Since the last 2 fields contain numeric data, lines 550 and 560 also convert the alphanumeric fields to numeric fields. Recall that in ASCII files, all data is in alphanumeric form.

Lines 690 through 750 provide the field width definitions for Multiplan using the SYLK format. Line 750 defines the 7th column in Multiplan which is to contain the product of the entries in columns 5 and 6. Note that the field widths for columns 4 through 6 are wider than the fields of the original data. Column 4 (line 720) was originally only 2 characters wide. Since the minimum column width in Multiplan is 3 characters, the field width had to be increased to 3. The remaining 2 columns are defined to be 1 character wider than the original data columns. That is no problem; the fields can be made as wide as one desires. For that matter, they can even be made narrower than the original data and still load properly into Multiplan. However,

ADAPTOR	MIL	UG175U	EA	1	.40
CABLE: COAXIAL RG58A/U	BELDEN	8259	FT	80	.329
CASE: UTILITY	HAMMOND	1411C	EA	1	1.02
CONNECTOR	AMPHNOL	126-806	EA	3	3.19
CONTROL	IRC	Q13133	EA	1	1.54

Figure 1

Figure 2

```
10    REM  PROGRAM ASCII2MP.BAS    R E HEATH      1986 JULY 12
20    CLS
30    PRINT "************************************************************
40    PRINT "* This  program  converts an  ASCII  data file into a SYLK *
50    PRINT "* (Symbolic Link) file which can  be  read  by  Multiplan. *
60    PRINT "* The  data  file  must  consist  of 4 alphanumeric fields *
70    PRINT "* whose widths are 24, 8, 16 and 2, followed by 2  numeric *
80    PRINT "* fields  whose  widths are 6 and 8.  If the fields in the *
90    PRINT "* ASCII data file are  different,  modifications  will  be *
100   PRINT "* required  in  program line 340, in program lines between *
110   PRINT "* 510 and 560 and in program lines between  690  and  900. *
120   PRINT "* Changes   may   also   be   required   elsewhere.    The *
130   PRINT "* alphanumeric  fields  in  the  source  ASCII  file  must *
140   PRINT "* contain  no commas, semicolons or tabs.  The source file *
150   PRINT "* can contain up to 255 lines.                             *
160   PRINT "************************************************************
170   PRINT ""
180   REM ****** DEFINITION OF VARIABLES AND LINE WHERE FIRST USED ******
190   REM
200   REM   A$(255,4) Array for alphanumeric fields.                  340
210   REM   X$(255,2) Array for numeric fields in alphanumeric form. 340
220   REM   X(255,2)  Array for numeric fields in numeric form.       340
230   REM   A$        Dummy variable for INKEY$                       330
240   REM   F$        ASCII input file name.                          380
250   REM   L$        ASCII input file line.                          480
260   REM   M$        Multiplan Symbolic Link output file name.       390
270   REM   I         Loop variable.                                  460
280   REM   J         Loop variable.                                  830
290   REM   M         Number of lines read from ASCII file.           590
300   REM   Y         Cursor line position.                           440
310   REM
320   PRINT "Press any key to continue."
330   A$ = INKEY$ : IF A$ = "" THEN 330
340   DIM A$(255,4), X$(255,2), X(255,2)
350   REM
360   REM ********************** INPUT FILE NAMES **********************
370   CLS
380   INPUT "Enter the name for the input ASCII file:"; F$
390   INPUT "Enter the name for the output SYLK file:"; M$
400   PRINT ""
410   REM
420   REM ********************** READ ASCII FILE ***********************
430   REM
440   Y = CSRLIN                      'Establish status line position.
450   OPEN "I", #1, F$                'Open input file named in line 380.
460   FOR I = 1 TO 255
470   IF EOF(1) THEN 580
480   INPUT #1, L$                    'Read a line of the file.
490   LOCATE Y, 1
500   PRINT "Reading line"; I; "of file "; F$ 'Provide status to screen.
510   A$(I,1) = MID$(L$,1,24)                 'Segment each record into
520   A$(I,2) = MID$(L$,25,8)                 ' fields for Multiplan.
530   A$(I,3) = MID$(L$,33,16)
540   A$(I,4) = MID$(L$,49,2)
550   X$(I,1) = MID$(L$,51,6) : X(I,1) = VAL(X$(I,1)) 'Convert from alpha-
560   X$(I,2) = MID$(L$,57,8) : X(I,2) = VAL(X$(I,2)) ' numeric to numeric
570   NEXT I                                          ' fields.
580   CLOSE #1
590   M = I - 1                       'Loop counted an extra line.
600   PRINT ""
610   PRINT "Reading of file "; F$; " is complete."
620   REM
630   REM **** INPUT OF ASCII FILE COMPLETE. START TO FORM SYLK FILE ****
640   REM
650   PRINT "Formation of Symbolic Link file "; M$; " now starting."
660   PRINT ""
670   OPEN "O", #2, M$                'Open output file named in line 390.
680   PRINT #2, "ID;PMP"              'First line for any SYLK file.
690   PRINT #2, "F;W1 1 24"           'Define width of fields.
700   PRINT #2, "F;W2 2 8"
710   PRINT #2, "F;W3 3 16"
720   PRINT #2, "F;W4 4 3"            'Minimum field width is 3.
730   PRINT #2, "F;W5 5 5"
740   PRINT #2, "F;W6 6 9"
750   PRINT #2, "F;W7 7 9"
760   PRINT #2, "F;FF3D;C6"           'Define decimal places for
770   PRINT #2, "F;FF2D;C7"           ' numeric fields.
780   PRINT #2, "B;Y255;X7"           'Define spreadsheet extent.
790   Y = CSRLIN                      'Establish status line position.
800   FOR I = 1 TO M
810   LOCATE Y, 1
820   PRINT "Line"; I; "of Multiplan SYLK file "; M$; " now being formed."
830      FOR J = 1 TO 4                         'Loop to form alphanumeric
840         PRINT #2, "C;Y";I;";X";J;";K";      ' entries in first
850         WRITE #2, A$(I,J)                   ' 4 columns.
860      NEXT J
870      FOR J = 5 TO 6                                 'Loop to form numeric
880         PRINT #2, "C;Y";I;";X";J;";K";X(I,J-4)      ' entries in last
890      NEXT J                                         ' 2 columns.
900      PRINT #2, "C;X7;ERC[-2]*RC[-1]"        'Formula for column 7.
910   NEXT I
920   CLOSE #2
930   PRINT ""
940   PRINT "Formation of SYLK file "; M$; " is complete."
950   END
```

in the latter case, overflow information would not be displayed in the Multiplan spreadsheet. Possible, but rather pointless.

The default cell format is assumed for the first 5 Multiplan columns. Lines 760 and 770 define a non-default format for columns 6 and 7, to show numerical data to 3 and 2 decimal places, respectively.

Line 780 defines the maximum spreadsheet extent, 255 rows by 7 columns. That line is actually not needed, but it is preferable to include it.

The nested loops from lines 800 to 920 are where the essential work takes place, the writing of the data from the ASCII file, now contained within BASIC array elements, into a SYLK file that can be read by Multiplan. An understanding of what these loops do, and why, is best accomplished by studying the result, the actual SYLK file formed, shown in Figure 3. The meaning of each line of the SYLK file can be deciphered by referring to Appendix A3 of the Multiplan manual. (If you are using a pirated copy of Multiplan and don't have a manual, my sympathy for you can readily be measured in a few angstrom units.)

Whether or not your ASCII data file has a format different than that of Figure 1, I recommend that you first create a data file identical to that of Figure 1 and use it for a test run to assure yourself that everything works properly. The procedure to use is as follows:

1. For simplicity of testing, copy the following files to a single disk, preferably a bootable disk:

```
ZBASIC.COM
MP.COM
MP.LOD
MP.SYS
MP.DAT
ASCII2MP.BAS
SAMPLE.DAT
```

2. Enter ZBASIC and load ASCII2MP.BAS.
3. RUN the program, entering SAMPLE.DAT for the input file at the prompt and whatever name you like, say SAMPLE.SYL, as the name for the output file at the prompt.
4. When the Ok prompt appears in ZBASIC, return to the operating system and enter Multiplan.
5. Enter Transfer Options Symbolic followed by Transfer Load SAMPLE.SYL. Multiplan will then read the SYLK file and end by displaying the data read on the screen.

ADAPTOR	MIL	UG175U	EA	1	0.400	0.40
CABLE: COAXIAL RG58A/U	BELDEN	8259	FT	80	0.329	26.32
CASE: UTILITY	HAMMOND	1411C	EA	1	1.020	1.02
CONNECTOR	AMPHNOL	126-806	EA	3	3.190	9.57
CONTROL	IRC	Q13133	EA	1	1.540	1.54

Figure 3

6. If you wish to save the new Multiplan worksheet in its normal form, enter Transfer Options Normal followed by Transfer Save SAMPLE.MP (or whatever filename suits you).

If anything doesn't work properly, it will be because of an entry error in either the input file SAMPLE.DOC or ASCII2MP.BAS. Clear up any of those problems before proceeding.

If your data is identical in format to that of Figure 1, you are now ready to process it in exactly the same way you did with SAMPLE.DAT. Otherwise, you will have to alter ASCII2MP.BAS to fit your particular data format. Read on.

Since a major purpose of this article is to help make it possible for you to revise my program to fit different data formats, I will now show the program revisions needed to fit two different input data formats.

Example 1

Suppose that the 4th field of your data format is a 6 character numeric field rather than the 2 character alphanumeric field that exists in my inventory data. Following is a listing of the revised program lines that would be needed to fit such a situation:

```
340     DIM A$(255,3), X$(255,3),
        X(255,3)
540     X$(I,1) = MID$(L$,49,6) :
        X(I,1) = VAL(X$(I,1))
550     X$(I,2) = MID$(L$,55,6) :
        X(I,2) = VAL(X$(I,2))
560     X$(I,3) = MID$(L$,61,8) :
        X(I,3) = VAL(X$(I,3))
720     PRINT #2, "F;W4 4 6"
830        FOR J = 1 TO 3
870        FOR J = 4 TO 6
880           PRINT #2, "C;Y";I;";X";J;
              ";K";X(I,J-3)
```

The change in line 340 reflects the need for a smaller alphanumeric array and a larger numeric array, since an original alphanumeric field has become a numeric field. The changes in lines 540 through 560 reflect the changes in the parsing of the input lines needed because fields 4, 5 and 6 now have different locations. Line 720 reflects the need to define a Multiplan field whose width is now 6 rather than the former 3. The indices on the two loops at 830 and 870 need to be changed to reflect the changes from 4 alphanumeric fields and 2 numeric fields to 3 and 3, respectively. The only change to line 880 is at the end, from J-4 to J-3, because the lower loop index has changed.

Example 2

Suppose that your data format has two fields of width 10 rather than the single first field of width 24 and both are alphanumeric fields. The following revised program lines would be needed to take care of that situation:

```
340     DIM A$(255,5), X$(255,2),
        X(255,2)
510     A$(I,1) = MID$(L$,1,10)
515     A$(I,2) = MID$(L$,11,10)
520     A$(I,3) = MID$(L$,21,8)
530     A$(I,4) = MID$(L$,29,16)
540     A$(I,5) = MID$(L$,45,2)
550     X$(I,1) = MID$(L$,47,6) :
        X(I,1) = VAL(X$(I,1))
560     X$(I,2) = MID$(L$,53,8) :
        X(I,2) = VAL(X$(I,2))
690     PRINT #2, "F;W1 1 10"
695     PRINT #2, "F;W2 2 10"
700     PRINT #2, "F;W3 3 8"
710     PRINT #2, "F;W4 4 16"
720     PRINT #2, "F;W5 5 3"
730     PRINT #2, "F;W6 6 5"
740     PRINT #2, "F;W7 7 9"
750     PRINT #2, "F;W8 8 9"
760     PRINT #2, "F;FF3D;C7"
770     PRINT #2, "F;FF2D;C8"
780     PRINT #2, "B;Y255;X8"
830        FOR J = 1 TO 5
870        FOR J = 6 TO 7
880           PRINT #2, "C;Y";I;";X";J;
              ";K";X(I,J-5)
900        PRINT #2, "C;X8;
           ERC[-2]*RC[-1]"
```

Many more program alterations are needed to take care of the situation, since an extra field has been added at the far left of each data line. Line 340 reflects the need for an extra column in the A$ array. Different locations for all the fields in the input data requires changes in all of the lines from 510 through 560, and the addition of line 515 to take care of the extra field. The changes in lines 690 through 750 reflect the changes in the definitions of the fields for the SYLK file, since an extra column must be added and all of the original fields from the second onwards shifted one to the right. The changes in lines 760 and 770 reflect the changes in the column numbers for Multiplan, since an extra column was added to the left of the numeric data columns. The changes to the loop indices in

lines 830 and 870 reflect the fact that an extra alphanumeric column has been added. The only change to line 880 is the change from J-4 to J-5, also to reflect the addition of an extra alphanumeric column to the left of the 2 numeric columns at the right. The only change in line 900 is from X7 to X8, because the column that was formerly number 7 now becomes number 8.

The two examples given above are not simply theoretical. Those revised program lines have actually been lifted from revised programs which were applied to the different data formats specified and found to work properly.

I've already made the point that the creation of the SYLK file with a BASIC program is better done without adding the programming complications needed to insert column headers. Those column headers can easily be inserted in the Multiplan file after it has read the SYLK file. But there's an even better way. Once the Multiplan file has been formed, write the spreadsheet to a file with the Print File command. (I recommend first setting the top and left margins to zero). Then, read that file into a word processor where any headings, including column headings, can be easily and quickly inserted with more flexibility than is available within Multiplan. It is especially easy to insert long header lines that can be quickly centered. If you had to replace any commas in the original data with some other character as I did, it would be necessary to bring that file into a word processor anyway to convert back to the commas. If one wants to do it the hard way, the colon, which I used, could be changed to a comma within Multiplan, since Multiplan itself allows commas within alphanumeric fields.

Multiplan does not read SYLK files with blazing speed. One of my data files, in the format shown in Figure 1, contained 179 lines. The SYLK file created by ASCII2MP .BAS consisted of more than 1000 lines. Multiplan required more than 4-1/2 minutes to read that file. For the first minute, only disk reading seemed to occur; then, the Multiplan status line showed which line of the SYLK file was being read, in much the same way as ASCII2MP.BAS provides status information on the screen. ASCII2MP.BAS creates a SYLK file much faster than Multiplan can read it.

If at some time in the future I again needed to create a lengthy data file, I believe I would again start with a word processor. By making use of tabs, data can be entered much faster than it can be entered into Multiplan cells and editing is much faster. Also note that the data in the final 2 columns of the original ASCII file, (Quantity and Unit Price), is not right justified, nor are any leading or trailing zeros included. Such formatting during data entry is not necessary. The required formatting is taken care of by the definitions included in ASCII2MP .BAS. That feature of data entry can save considerable time in the preparation of a long listing when using a word processor. Now that I know how easily such ASCII data files can be converted into SYLK files and read into Multiplan, the word processor to Multiplan route should save considerable time in the long run. I recommend it. On the other hand, updating an existing Multiplan file with minor changes or additions is better done within Multiplan itself.

```
ID;PMP
F;W1 1 24
F;W2 2 8
F;W3 3 16
F;W4 4 3
F;W5 5 5
F;W6 6 9
F;W7 7 9
F;FF3D;C6
F;FF2D;C7
B;Y255;X7
C;Y 1 ;X 1 ;K"ADAPTOR                    "
C;Y 1 ;X 2 ;K"MIL      "
C;Y 1 ;X 3 ;K"UG175U            "
C;Y 1 ;X 4 ;K"EA"
C;Y 1 ;X 5 ;K 1
C;Y 1 ;X 6 ;K .4
C;X7;ERC[^2]*RC[^1]
C;Y 2 ;X 1 ;K"CABLE: COAXIAL RG58A/U "
C;Y 2 ;X 2 ;K"BELDEN   "
C;Y 2 ;X 3 ;K"8259              "
C;Y 2 ;X 4 ;K"FT"
C;Y 2 ;X 5 ;K 80
C;Y 2 ;X 6 ;K .329
C;X7;ERC[^2]*RC[^1]
C;Y 3 ;X 1 ;K"CASE: UTILITY             "
C;Y 3 ;X 2 ;K"HAMMOND "
C;Y 3 ;X 3 ;K"1411C             "
C;Y 3 ;X 4 ;K"EA"
C;Y 3 ;X 5 ;K 1
C;Y 3 ;X 6 ;K 1.02
C;X7;ERC[^2]*RC[^1]
C;Y 4 ;X 1 ;K"CONNECTOR                  "
C;Y 4 ;X 2 ;K"AMPHNOL "
C;Y 4 ;X 3 ;K"126^806           "
C;Y 4 ;X 4 ;K"EA"
C;Y 4 ;X 5 ;K 3
C;Y 4 ;X 6 ;K 3.19
C;X7;ERC[^2]*RC[^1]
C;Y 5 ;X 1 ;K"CONTROL                    "
C;Y 5 ;X 2 ;K"IRC      "
C;Y 5 ;X 3 ;K"Q13133            "
C;Y 5 ;X 4 ;K"EA"
C;Y 5 ;X 5 ;K 1
C;Y 5 ;X 6 ;K 1.54
C;X7;ERC[^2]*RC[^1]
```

Figure 4

The central problem that I've addressed is that of converting a lengthy ASCII data file into a form that can be read into Multiplan. The task is simplified by concentrating on the data and forgetting about headers of any kind which can be added later. I believe that the program ASCII2MP.BAS can be used as it stands, or modified by most readers with the guidance and examples I've provided, to fit almost any likely data format. I would welcome word from anyone who is able to apply the program, modified or not, to meet some application. Because of very limited time through most of the year, it's not likely I could provide much help to anyone unable to modify the program to fit some special need.

In closing, I can't resist the inclusion of a few comments about word processors. Easywriter II is one word processor that cannot be used for the purposes I've suggested, since it neither reads nor writes ASCII files. Fortunately, as far as I know, Easywriter II is not available for the H/Z-100, a great blessing in my opinion, since I rate it as the worst word processor I've ever used. May it stay with the IBM PC where it belongs. WatchWord is my favorite by a wide margin. It is very powerful and flexible, particularly because of its configuration capabilities and extensive macro language. It supports a wide variety of printers and can be configured to take advantage of any feature of any printer. And WatchWord is FAST. The only significant limitations I've encountered are that it will handle only single line headers and footers and has no footnote capability. If you can live with those limitations, I highly recommend WatchWord as the best software bargain available. Either WordStar or PeachText could also be used for the purposes I've suggested, though I don't think much of the latter which I struggled with for more than a year as my only word processor before buying WatchWord. There could be a problem with WordStar's high order bits; I haven't investigated that possibility.

For the record, I used the following hardware and software in the preparation of ASCII2MP.BAS:

H-120-1 Computer (5 MHz)
  w/ dual full-height Tandon drives.
H-207-40 Dual half-height Schugart 8
  in drive accessory.
MS-DOS Version 2.21
ZBASIC Rev. 1.0
Multiplan Version 1.02
WatchWord Version 2.08

*

FINAL Z-100 SOFTWARE CLOSE OUT

Zenith packages with software, manuals and registration cards for the original Z-100 computer series (not for the IBM compatibles).

PART NUMBER	DESCRIPTION	LIST PRICE	SALE PRICE
MS-463-1	Z-Basic (16 bit) interpreter	$175.00	$10.00
MS-463-7	Multiplan	$195.00	$10.00
CB-463-11	Z-Chart	$150.00	$10.00
CD-463-2	Condor File Manager	$299.00	$10.00
PK-100-4	All 4 listed above	$819.00	$38.00
MS-253-1	Microsoft BASIC-80 (8-bit)	$175.00	$10.00
OS-53-2	CP/M-85 (8 bit)	$150.00	$15.00
OS-63-4	Z-DOS	$150.00	$20.00
CB-463-9	PECON Peachtree to Condor	$99.00	$15.00
RS-463-5	Peachtree Inventory Management	$499.00	SOLD
WI-463-1	Remote Batch Terminal Emulator	$899.00	SOLD

★★★ UPGRADE ACCESSORIES FOR Z-100 ★★★ SERIES COMPUTERS

HIGH DENSITY 1.2 MEG DRIVES. External floppy drive set-up complete with drive, power supply, case and cable. Ready to connect to your 8" floppy controller. $277.00. Dual Drive Unit $424.00

COLOR GRAPHICS UPGRADE. All memory chips (16 pieces 150 ns) required to update Z-100 Series computer for color. Installation instructions included. Order Memory Kit #100-64-16 .....$21.50

MEMORY UPGRADE. All memory chips (9 pieces 150 ns) required to upgrade from 128K to 192K RAM. Installation instructions included. Order Memory Kit #100-64-9 .....$12.50

ZMF100A by FBE Research. A modification package which allows 256K chips to be used on the old-style motherboard to reach 768K. Simple assembly with no soldering or trace cutting. Compatible with Easy PC and Gemini Emulator. $60.00 alone or $148.50 with 27 256K RAM chips included.

SmartWatch by FBE Research. If you don't have a clock for your Z-100, get this one. More details under Z-150 upgrade listings $44.00

GEMINI EMULATOR BOARD. Makes the Z-100 compatible with the IBM PC library of programs $432.00

UCI EASY PC. IBM PC Emulator. Makes your Z-100 IBM Software Compatible. Full 8 MEG operation, color graphics and audio compatible. Retail $699.000, Payload $477.00

UCI EASY 87. Add an 8087 Numeric Coprocessor. $69.00 for the board without an 8087 Chip. With 5 MEG 8087 $197.00 or with 8 MEG 8087 installed $234.00

UCI MEMORY UPGRADE CARD. We recommend this one highly. The board has sockets for up to 2 MEG of RAM. With no RAM installed $328.00. With RAM installed and fully tested, 512K $387.00, One MEG $446.00, Two MEG $564.00 Add $35.00 for EasyDrive RAM Drive Software if desired.

UCI RAMSAVER. Maintains power on UCI MEMORY CARD RAM when computer is off. Save your programs in RAM with your computer off. Payload $177.00

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

UCI EasyWin. Winchester Drive Systems at reasonable prices. Complete Hard Disk Systems for mounting inside your Z-100. Systems complete with Seagate Drives, 21 MEG $578.00, 31 MEG $598.00
System without Drive and Controller $239.00

CDR Z-100 SPEED MODULE. Run your Z-100 Computer at 7.5 MHz. Installs easily with no soldering. Externally switchable between Speed and Normal mode. Payload $48.00

*** PANASONIC DOT MATRIX PRINTERS ***

KPX-10801 120 cps 10" $219.00
KPX-10911 160 cps 10" $272.00
29cps NLQ
KPX-10921 180 cps 10" $338.00
33 cps NLQ
KPX-1592 180 cps 16.5" $421.00
38 cps NLQ
KPX-1595 240 cps 16.5" $462.00

★★★ UPGRADE ACCESSORIES FOR Z-150/160 ★★★ SERIES COMPUTERS

SmartWatch from FBE Research. Installs in ROM Socket on CPU Board in Zenith computer series Z-100/150/158/160. This tiny jewel of a product contains a ten year battery and keeps your computer informed of both time and date at each boot-up. Complete instructions and software included $44.00

MEMORY KIT #150-256-18. Includes a ZPAL chip which allows use of 256K RAM chips included (18 pieces 256K 150 ns RAM chips). Kit increases 128k memory to 640K or 256K memory to 704K. All chips plug into your existing Zenith Memory Board. Unbelievable but true $87.00

Winchester Hard Disk Drive Internal Set-up. Includes Winchester drive, controller/interface card, cables and all hardware. With 20 MEG (formated) drive $339.00 30 MEG $359.00. May be installed in Z-148 using an Expansion Card sold below.

PTZ-148 Expansion Card for Z-148. Includes 2 expansion slots plus a clock/calendar. $118.00

Winchester Hard Disk Drive External Set-up. Includes Winchester drive, controller/interface card, power supply and case with fan.
With 20 MEG (formated) drive $528.00
With 30 MEG (formated) drive $548.00

ZFL-181-93 LAPTOP PORTABLE with the amazing Supertwist LCD backlit screen 640K RAM, two 720K 3.5" disk drives, clock, P & S Ports, 8 MEG. Retail $2399.00. Payload $1739.00

★★★ HALF HEIGHT FLOPPY ★★★ DISK DRIVES

MITSUBISHI M2896	8"	48 TPI DS/DD	1.2 MEG $375.00
MITSUBISHI M501	5.25"	48 TPI DS/DD	320K/360K $105.00
MITSUBISHI M504	5.25"	96 TPI DS/DD	360K/1.2 MEG .$152.00

★★★ SEAGATE HARD DISK DRIVES ★★★

ST-225 20 MEG Winchester Hard Disk $295.00
With Western Digital Controller & Cables $339.00
ST-238 30 MEG, Requires RLL type controller $309.00
With RLL Controller & Cables $359.00
ST-4038 30 MEG High Speed for Z-200 ..... $544.00
ST-4051 40 MEG High Speed for Z-200 ..... $629.00
ST-4096 80 MEG High Speed with Software $929.00
ST-251 40 MEG High Speed Z-150/200 .....$489.00

★★★ POWER SUPPLIES AND CASES FOR DISK DRIVES ★★★

Rugged steel construction with heavy duty power supplies. Purchase with drives and we will install drives in case.

Single 5.25" unit for Full Height Drive $68.00
Dual 5.25" unit, for Half Height Drives $92.00
Dual 8" unit with fan, for Half Height Drives $188.00
Face plate for single drive in dual case $8.00
For WINCHESTER Drive, with fan $191.00

★★★ CHIP SPECIALS ★★★

The finest RAM available and at PAYLOAD prices. Order one to one thousand chips and add only $2.00 for shipping.

64K Dynamic RAM, 150 ns .....$1.35 each
256K Dynamic RAM, 150 ns ..... $3.28 each
256K Dynamic RAM, 120 ns ..... $3.87 each

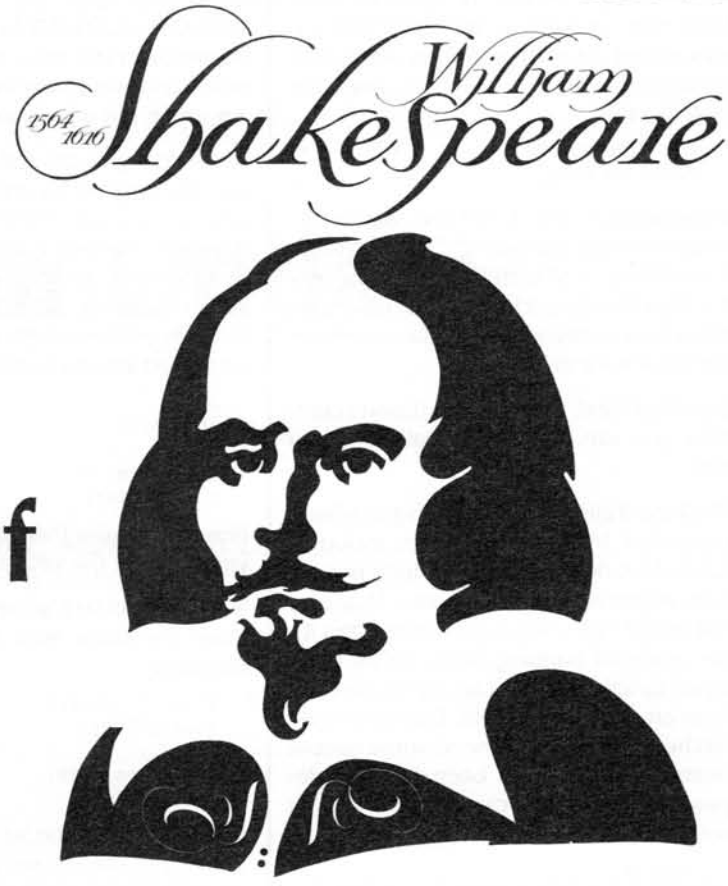
V-20 CHIPS. High Speed NEC V-20-8 8088 replacement. These run at up to 8 MEG and are said to increase CPU speed 10-30%.
Payload $14.75

8087 MATH COPROCESSOR CHIPS. Speeds and improves numeric processing. 5 MEG 8087-3 .... $129.00, 8 MEG 8087-2 .... $165.00

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

Please **MAIL** or **PHONE** your order today and expect prompt service. **MASTERCARD** and **VISA** gladly accepted with no additional charge. All hardware carries a 90 or more day warranty. Add $5.00 to all prepaid orders for handling and shipping, we pay the balance. Texas Residents please add 7.25% sales tax. We accept purchase orders from schools, government and approved accounts.

To scanf Or Not To scanf



Don Keller
1330 Eden Valley Road
Port Angeles, WA 98362

The presence or absence of scanf — the formatted input function — is a kind of measure of how complete a particular version of the C language may be. Certainly, there are plenty of other considerations, but you have to have scanf to play with the big boys. One reason for scanf's prestige could be the fact that it's about the most expensive function you're likely to have in terms of the demands it makes on the rest of the system, including the programmer. Is this giant really necessary? Most of the time I'd have to say no. Too bad I can't say it's never necessary and forget about using it, but that's not the case; it will do some tricks that you might not want to get along without. If you are going to use it, take all possible advantage of documentation you already have.

Assuming you use C/80 from The Software Toolworks (I'd like to expound on other compilers too, but I don't have any other compilers), you have a fair amount of information on scanf. There's more than just the description in the manual — quite a bit more. The source code listing of scanf begins with a full page of comments describing C/80's version of formatted input. You should also have a copy of "The C Programming Language" by Brian W. Kernighan and Dennis M. Ritchie, published by Prentice-Hall in Englewood Cliffs, New Jersey 07632. Any non-trivial purveyor of computer related books can get it for you if you don't already have it. The C/80 manual says K&R is the final authority on C/80; in fact, the appendix to "The C Programming Language" defines C. I've seen the opinion expressed in print several times that K&R is pretty heavy reading and not suitable for beginning C programmers. I wouldn't recommend it for the textbook in an elementary school computer class, but C programmers need it — newcomers most of all. Learn to hunt through it for solutions to problems as you encounter them. You don't have to read K&R like a novel. You don't need to memorize it either; but if your copy doesn't take on a slightly battered and dog-eared look, maybe you ought to find an improved BASIC interpreter. It's probably a good idea to work as many of the examples as you can. Some of them are useful utilities. A discussion of scanf begins on page 147.

There are at least three common traps in scanf for unwary programmers: It looks a lot like printf — specification strings are quite similar — but scanf DOES NOT ASSIGN VALUES TO VARIABLES. It assigns values to memory locations POINTED TO by identifiers in the argument list.

```
int n;

scanf("%d", n);
```

WON'T WORK!

```
int n;

scanf("%d", &n);
```

works because we supplied the address of n.

```
int *n;

scanf("%d",  n);
```

also works. This time n is a pointer to an int and we can access the value stored there as *n (the content of n).

The other two killers are peculiar, but not necessarily unique, to C/80. scanf does return a value and if you want to assign that value to a variable, you'll need some extra parentheses, like this:

```
int n, rv;

rv = (scanf("%d",  &n));
```

assigns the input value to n, and the value returned by scanf to rv. The returned value is the number of items matched and assigned — in this case, 1 — or End Of File, if that was encountered.

Finally, there's a potential problem that has to do with organization. Grant Gustafson

wrote a single version of scanf to serve C/80, with or without the MATHPAK enhancement for 32-bit numbers. At the top of the source file you'll find two lines that look like this:

```
/*#define NOLONG          /* . . . */
/*#define NOFLOAT         /* . . . */
```

If you do NOT have MATHPAK, delete the initial slash and asterisk, "/*," on both lines. If you DO have MATHPAK, leave it the way it is. You have to get this right unless you're trying to set a new record for the number of compiler error messages.

Besides all that, scanf is big, whatever compiler you use. There is usually a better way.

The best of all input functions is getc(chan), supported by most compilers, including C/80. All it does is fetch one byte per call from a specified input channel. That may not sound especially sophisticated, but it's the essential building block for all other input functions. You can input anything from anywhere with getc. Each byte getc fetches may very well need some special treatment after it has been fetched, depending on where it came from and what you want it for.

```
char c;

c = getc(stdin);
```

assigns to C the next byte available from the standard input. stdin was #defined as the standard input in a header file; take a look at the last part of section 8 in the C/80 manual. The standard input will be the terminal unless redirected in the command line or reassigned somewhere in the running program. Instead of stdin, we could use a channel number provided by the fopen function to get input from a disk file or a device driver for an alternate terminal, a MODEM, or whatever. By the way, getc (stdin) is the precise equivalent of getchar().

At the next higher level, getline uses getc (stdin) to read a string of characters from the standard input and store them at a specified location in memory.

```
char s[100];

getline(s, 100);
```

will accept as many as 99 characters, leaving the last byte for a terminating '\0,' which is tacked onto the end of the string by getline, and store the string at the location identified by 's.' There are some limitations here; only the standard input can be read, and getline does not explicitly signal End Of File. To avoid these shortcomings, some version of fgets is often substituted for getline. (See my article, "C/80 in OMDOS, HDOS 3.0 [and M-80]" in the December 1987 issue of REMark for one version of fgets.) Now we can get text from any input, but what about numbers?

Numeric digits are just characters with a specific purpose. If a text string represents a number, it needs a little special processing to yield its numeric value. That's what functions like atoi are for. Type in a signed integer (-32768 to 32767) at your keyboard through getline and the resulting string can be turned into an integer value by atoi:

```
int n;
char s[7];

getline(s, 7);
n = atoi(s);
```

Now n contains the value of the number you typed at the keyboard.

With the MATHPAK added to C/80, atof does the same trick for floating point numbers:

```
float f, atof();
char s[100];

getline(s, 100);
f = atof(s);
```

Now f contains the value of the number you typed. The text representation of a real number could be quite large so we allowed room for 99 characters in the array, 's.' Notice that since atof returns a float, it must be declared float. Otherwise, the compiler will assume it returns an int which can cause premature baldness. So far, so good; we can get text, integers, and real numbers from anywhere. That leaves longs and numbers expressed in octal and hexadecimal.

ASCII to numeric — aton.c is a collection of conversion functions to make it possible to input 16 and 32 bit integers in decimal, octal, and hexadecimal. They all work on a string collected by getline or fgets. Come to that, you could also fill a character array with getc or getchar. First, you need an array:

```
char ns[length];
```

'length' should be 7 for decimal ints (five digits, plus an optional sign, plus room for the terminating null byte), 12 for decimal longs (same thing, except for starting with ten digits). Octals may be longer and hexadecimals shorter. A float is arbitrarily long and probably ought to be the length of an input line, say 80 or 100.

```
char *pn;

pn = ns;
while ((*pn = getc(wherever)) != '\n'
    && *pn++ != -1)  ;
*pn = '\0';
```

Assigning the address of the array, 'ns,' to the pointer, 'pn,' gives us a guaranteed pointer to work with instead of indexing the array. It really is easier. The while loop keeps assigning one character after another to the string until either a newline or End Of File is found. "*pn = '\0';" puts the null byte on the end of the string so that any normal C function can find the end of the string. 'wherever' is the number of the input channel; it could be 'stdin,' the standard input or it could be the number of a channel opened for read by fopen. Now that we have a string — one way or another — let's see how we can extract the numeric value it represents.

qtou(s) — Assumes the string, 's,' represents an unsigned 16-bit octal number. The returned value should be assigned to a variable name which was declared unsigned. Conversion stops when a character is encountered which could not be a digit in base eight, and the value constructed up to that point is returned. That means that in addition to mistaken entries, a carriage return or End Of File will terminate the conversion process.

xtou(s) — Works exactly like qtou, except that the string is assumed to represent a hexadecimal number. Acceptable digits are: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a or A, b or B, c or C, d or D, e or E, f or F.

long atol(s) — Converts integers in the range -2,147,483,647 to 2,147,483,647. I know the C/80 manual says long integers go down to -2,147,483,638, but they sure don't on my system. Notice that the function itself has to be declared long. Without that 'cast', atol wouldn't give us a thing we don't already get from atoi. The value returned needs to be assigned to a variable name declared long.

long qtol(s) — Works just like qtou above, but for 32-bit numbers.

long xtol(s) — Works just like xtou above, but for 32-bit numbers.

You might be able to find enough rope to hang yourself with unsigned 32-bit numbers here. C/80 doesn't support unsigned longs, but you can type one in with either qtol or xtol and print it back out again with fprintf. For instance, you could assign FFF FFFFF to a long variable with xtol, print the variable with %lx in fprintf and get the FFFFFFFF back. In the meantime, though, the compiler stored it as -1. Juggle eggs if you like, but bear in mind you might drop one from time to time.

aton looks a bit large; when I cooked it down to a .REL file, it turned out to occupy

seven sectors. Since that means it uses eight sectors on the disk, and scanf.rel needs twelve sectors, the saving in space doesn't amount to much. Combine that with the room taken by some version of fgets and the saving disappears. So why bother?

Well, it's like this: All of aton's utilities are in small individual functions; only those you actually call get loaded. scanf comes in one BIG chunk. If you're working without MATHPAK you can delete everything in aton, except qtou and xtou. It'll be much smaller without the 32-bit functions. You should never have to load all of the functions in aton. For instance, if you plan to input both 16-bit and 32-bit hexadecimal numbers, convert both sizes with xtol and don't call xtou. The 32-bit functions will assign 16-bit values to 16-bit variables just fine, so there's no need to use both kinds in one program. Overall, using aton makes for smaller programs. Also, scanf doesn't like character mode input from the console.

I still don't mean to belittle scanf's worth — far from it. You can collect multiple input objects with a single scanf call and they need not be all the same type. You can also be very selective about what parts of an input stream you want. Those are not trivial abilities, but I find that I seldom need them. scanf is a delight to have when you really need it; in the meantime, leaner functions will serve.

Listing

```
/*      aton.c          convert ASCII string to numeric value
 */
qtou(s)         /* ASCII octal to unsigned (16 bit) */
char *s;
{
        int n;
        n = 0;
        while (1) {
                switch (*s) {
                        case '0': case '1': case '2': case '3': case '4':
                        case '5': case '6': case '7':
                                break;
                        default: return n;
                }
                n = n * 8 + *s++ - '0';
        }
}
xtou(s)         /* ASCII hexadecimal to unsigned (16 bit) */
char *s;
{
        int n;
        n = 0;
        while (1) {
                switch (*s) {
                        case '0': case '1': case '2': case '3': case '4':
                        case '5': case '6': case '7': case '8': case '9':
                                break;
                        case 'a': case 'b': case 'c': case 'd': case 'e':
                        case 'f':
                                *s = *s - 'a' + '0' + 10;
                                break;
                        case 'A': case 'B': case 'C': case 'D': case 'E':
                        case 'F':
                                *s = *s - 'A' + '0' + 10;
                                break;
                        default: return n;
                }
                n = n * 16 + *s++ - '0';
        }
}
```

```
long atol(s)    /* ASCII decimal to long (32 bit) */
char *s;
{
        long n, sign;

        n = 0;
        sign = 1;
        while (1) {
                switch (*s) {
                        case '0': case '1': case '2': case '3': case '4':
                        case '5': case '6': case '7': case '8': case '9':
                                break;
                        case '-': ++s; sign = -1; continue;
                        case '+': ++s; sign = 1; continue;
                        default: return n * sign;
                }
                n = n * 10 + *s++ - '0';
        }
}
long qtol(s)    /* ASCII octal to long (32 bit) */
char *s;
{
        long n;

        n = 0;
        while (1) {
                switch (*s) {
                        case '0': case '1': case '2': case '3': case '4':
                        case '5': case '6': case '7':
                                break;
                        default: return n;
                }
                n = n * 8 + *s++ - '0';
        }
}
long xtol(s)    /* ASCII hexadecimal to long (32 bit) */
char *s;
{
        long n;

        n = 0;
        while (1) {
                switch (*s) {
                        case '0': case '1': case '2': case '3': case '4':
                        case '5': case '6': case '7': case '8': case '9':
                                break;
                        case 'a': case 'b': case 'c': case 'd': case 'e':
                        case 'f':
                                *s = *s - 'a' + '0' + 10;
                                break;
                        case 'A': case 'B': case 'C': case 'D': case 'E':
                        case 'F':
                                *s = *s - 'A' + '0' + 10;
                                break;
                        default: return n;
                }
                n = n * 16 + *s++ - '0';
        }
}
```

An MBASIC Utility ASCII Text Files Into "QUERY!3"

Kirk L. Thompson
#6 West Branch Mobile Home Village
West Branch, IA 52358

In the last installment of this erratic series (REMark, Dec., 1986), I observed that when you transfer data files to other computer systems, you should adapt the source file to the destination system as much as possible before the transfer. But there are also times, on one system, when you wish to convert one type of file to another. Though not "MAILPRO"-related, as are the others in this series, I will describe a utility, here, that will convert a standard, sequential, ASCII file into one readable by Hoyle and Hoyle's database manager, QUERY!3.

QUERY!3

To plug Hoyle and Hoyle's product for a moment, this package is an outstanding value, no matter which operating system version, HDOS, CP/M-80, or MSDOS, you obtain. It is undoubtedly the most powerful database package you can buy for HDOS. And while there are more capable programs available for CP/M and MSDOS, you also must pay much, much more for them. Moreover, if you purchase the optional report generator, QUERY!3 provides you with power equivalent to database managers costing two to three times its price. But getting back to the conversion utility . . .

TXT2Q3 Overview

The most universally recognized file type for both 8- and 16-bit computers is ASCII-sequential. Many programs, including "MAILPRO," QUERY!3, Lotus 1-2-3, and of course, text editors, can create this file type. But up till now, there has been no way of converting this universal type to one readable by QUERY!3. The MBASIC program in Listing 1, TXT2Q3.MBS, corrects this deficiency.

This program was designed for use under all three operating systems available to the Heath/Zenith computer community, with a mininum of modification on your part. While the program was developed under HDOS MBASIC 4.82, it does not use any special terminal codes. Nor does its code use special features unique to a particular version of Microsoft's interpreter. (The only very minor exceptions are the use of the at sign (@) to mark the end of physical lines within logical lines and the values of Boolean "true" and "false" [-1 and 0, respectively], as required by the HDOS version.)

But TXT2Q3 also has to be flexible enough to write the proper format for each version of QUERY!3. It accomplishes this by doing things which are usually considered the domain of assembly language programming, not BASIC.

Configuring A BASIC Utility?

For this utility, the format differences between the three disk operating systems boil down to one, and only one, factor: the number of bytes per sector. HDOS has 256; MSDOS uses 512; and CP/M-80 can have these values, plus 128 or 1024 bytes per sector, depending on disk formatting. QUERY!3, you see, is a sector oriented program package. It expects the data in the file to start at the beginning of the sector that follows the file definition sector(s). And this difference in sector sizes is a hurdle which TXT2Q3 must overcome to be usable under all three operating systems.

The obvious way around the problem is to make the utility configurable and "semi-intelligent." These actually go hand-in-hand. You, the user, must include a value in the code which represents the size of the sectors on your system. Once in the code, you can forget about it.

Actually, TXT2Q3 requires that you enter two values. Look at line 480 in the listing. Here, you see two variables, RBSIZ and SSIZ. The latter takes the value of the sector size on your system; the value in the listing is that for HDOS. The other variable is necessary because the utility must somehow write back the size of the data file after processing is complete. It does this by re-opening the completed file for random access and writing to two bytes near the beginning of the file, where this is stored. But to do this without you having to specify the size of the random buffer with MBASIC's random buffer size switch (/S:) under CP/M and MSDOS, you must include the default size of this buffer for your BASIC. Normally, this will be 128, except for HDOS, which takes the value in the listing.

After you have entered values for RBSIZ and SSIZ, the program then makes a quick check for typographical errors in the latter. This is done by determining the modulus of that value and 128; if this calculation is other than zero, the program promptly terminates with an error message.

Following that, on the next line of code, another decision is made, based on the sector size you assigned SSIZ. The problem here, is ensuring enough space for a reasonable number of field definitions in the data file "header" sector. This version of TXT2Q3 has a limit on the maximum number of fields it can accept for a data file. These are directly dependent on DOS sector size, since the program normally allocates only one sector for this. For HDOS,

For Converting Hoyle and Hoyle's Database File Format

this maximum is 14; for MSDOS, it is 29; but for CP/M, it could vary from 7 to 59. And I deemed the lowest value as insufficient for many practical applications.

So line 490 corrects this situation by allocating another sector for field information on those CP/M systems which have 128-byte physical sectors. It does this by offsetting the sector number in the datafile where the data actually starts (controlled by variable DSS) by one, thus reserving two sectors, for a maximum of 14 fields, in the "header". As I mentioned, all other systems reserve only one sector for field information.

Writing The File

Now what I have discussed above is merely set-up. These were the decisions, mostly design, which I considered. But actually writing the QUERY!3 format destination file borders on the traditional preserve of the assembly language programmer. The reason is that QUERY!3 data files are binary, as I shall demonstrate!

Line 1050 writes the first six bytes of the data file. These values tell the program package how many fields there are in each record (byte 1), the length of a record (bytes 2 and 3), the sector in the file where the data actually starts (byte 4 and the value of variable DSS in TXT2Q3), and the number of records in the file, plus one (bytes 5 and 6). You will note that these are written with BASIC's "CHR$()" function and that is because these are binary numbers. In fact, the two-byte values are also in Intel's low-byte/high-byte sequence, which requires figuring the modulus and integer quotient of the record length in this line. (A dummy value of 0001 hex is written for the two-byte file length since the program does not know what that is, yet.)

At the end of this line, you will observe the presence of a byte counter (BCT!). Updating this counter at every write is crucial to the proper format of the destination datafile. You will also notice that it is declared as a single-precision variable. All other numeric variables are predefined as integers by the "DEFINT A-Z" statement in line 470 for fast execution. Of course, had BCT been included in the integer declaration, files larger that 32K could not be written because the MBASIC interpreter places that limit on integers.

Lines 1070 and 1080 write the length of each field and the field label; again, the "CHR$()" function is used to create a binary number in the file for the former. And each field label is padded out between its last character and the 16-character maximum length which QUERY!3 permits, by a "FOR...NEXT" loop with the undefined ASCII character, 128 decimal (80 hex).

At line 1110, with the field lengths and labels written, the remainder of the "header" sector is filled in with the ASCII 128 decimal, this time all the way to the sector boundary.

This same padding also takes place after the last of the actual data is written, as you can see in line 1360. But in this instance, I am taking the easier way out; QUERY!3 typically dumps junk strings out here to fill out the last sector in the file. It was easier for me to simply use its ubiquitous 80 hex!

Padding with this same character also occurs between the end of the source fields and the length for the field you specified for the destination file. This is done with another "FOR...NEXT" loop at the end of line 1080.

Finally, the last binary write is performed, in lines 1420 and 1430, when the length of the file is written back to the "header" sector with a random access. Since this two-byte number is in Intel low/high format, it must be tweeked with the modulus and integer division operators. But thus far, I have only discussed some of the nitty-gritty internal workings of the utility. The thing for you to do now, is type it in (more on that below).

Preparing To Use The Utility

Once you've adapted it to your Microsoft BASIC and plugged in your values of SSIZ and RBSIZ discussed above, you are ready to use TXT2Q3. But before you can do anything, you must have a sequential file of data to convert to QUERY!3 format. You could use a text editor to create one. In my case, I am moving some of my "MAILPRO" files to QUERY!3 and use the "LABELS" sub-program of the former to dump the data to disk. I know of one person who, with an idea that can only be termed inspired, uses the "Create Standard Disk File" output option in QUERY!3's "SEARCH" program to create a sequential file on his H-100 under ZDOS. Then he edits it with an editor and uses my original version of TXT2Q3 to convert it back to Hoyle and Hoyle's database format. (You will note from the listing that what I am presenting here is version 1.2.) But the format of this ASCII sequential file is somewhat critical.

Figure 1 illustrates a typical application you might have. This is an address-list file, using the address-label field layout recom-

Textfile source fields	QUERY!3 destination Field label	Length
Joe	First name	20
Adams	Last name	30
	Business name	30
3934 Zerzes Ave N	Address	30
Minneapolis	City	30
MN	State	2
55412	ZIP	5
612-123-4567	Phone number	12
(optional blank line, see text)	(skipped)	
Kirk L.	First name	20
Thompson	Last name	30
	Business name	30

Figure 1
Example Source And Destination File Layouts

mended in Hoyle and Hoyle's documentation. The fields must be in this order, but the lengths are at your discretion. On the right are the field labels and lengths which the converted file will have. The format of the sequential file is shown on the left. Notice that the lines in the source file are one-for-one with the fields in the destination file. Also note that all of the sequential fields are left-flush; if they aren't, the fields in the destination will begin with spaces. Finally, observe the lines near the bottom of the figure in parentheses.

TXT2Q3 gives you the option of including a "blank" line between records in your source file. During processing, you are asked if the file contains these and if it does, they are skipped when the data is moved to the destination. While I originally conceived this option as providing a method of demarcating records in the source, you could use them for notes or record numbering. Whatever is in them will not appear in the destination datafile.

Running TXT2Q3

So now that you have a source file, load TXT2Q3 into your MBASIC interpreter and run it. After the program signs on, you are prompted for the names of the source and destination files. After you verify the name of each, the program looks for the files. If the source file is there, you are prompted for the destination. If the program cannot find the source, you are reprompted for its name.

When keying in the destination file name, do not include any extension. The default is QUERY!3's default, ".DTB". If you so much as include a period in this name, you will be reprompted for it. You will also be reprompted for the destination if the program finds a file with the same name. This prevents you from destroying data you might already have.

Then, the utility branches to a subroutine which is the major difference between this and earlier versions. This routine looks for a file on the same disk as your source file and with the same name, but a ".CRE" extension. This file is normally transparent to you, the user. It contains the destination file field definitions so that you do not have to manually key in this information each time you run the utility on the same source file. But since this is your first time through, the absence of the file is trapped and the error message, "!--> File not found!", is displayed on the terminal screen.

At this point, if the specification file was there, you could completely skip the manual field entry routine, but since it is not, you must. It really isn't all that bad. In fact, it closely emulates QUERY!3's "CREATE" program. Like that program, you are prompted for the number of fields you wish for the file and then their lengths and labels. The only thing TXT2Q3 lacks is a built-in field edit routine, so be careful when you key in this information. When finished, you have the option of saving or aborting the present entries. If you abort, you are reprompted for the field information.

Following entry of the fields, you are asked whether your source file contains blank lines between records. Whether you respond with "Y" or "N," the field specification file is then automatically created and the data you just typed in written to it. It will be waiting for you the next time you reconvert the source file after editing or whatever. Next, the files are opened, and reading and writing between source and destination begin in earnest. But one thing you will want to check is whether the data in your source file is correctly transferred to the latter. TXT2Q3 permits this since it will display the first and every tenth record on the terminal screen for your verification. If you note any problems, merely respond "N" to the "Are these OK (Y/N)<Y>?" prompt below the input record display and the utility will abort its processing and close the files.

If you have a problem, check your source file first. This is the most likely cause of errors if you keyed in the BASIC code correctly.

After all of the data in the source file has been transferred to the destination, the files are closed. Then, as I noted above, the destination is re-opened as a random file and the number of records in the file written to the "header." Finally, TXT2Q3 displays a wrap-up of its activity. It tells you the number of data records processed, the number of source records with field lengths longer than those you keyed in when you specified field lengths for the destination, the total bytes written, and the number of sectors the program wrote.

After the conversion is complete, I suggest examining the destination file with QUERY!3's "VIEW" program to ensure that it is as it should be. This is a good debugging aid if you run into problems; if you see garbage, something is obviously wrong!

Support

As with the other utilities I am publishing in REMark, if you have trouble, let me know. I would be happy to assist you; just write to the address above, or in the source code.

If you do not want to key in the latter, you also have a couple of alternative methods for getting it. You can send me a formatted, HDOS or CP/M-80, 40-track, hard- or soft-sector disk with $4 and I would be pleased to copy the files for you, including extensive documentation. If you include a stamped, self-addressed, return mailer, my charge is $2.

On the other hand, particularly if you are running MSDOS, you can download source (in ASCII format) and documentation from HUG's own PBBS. If you are adapting TXT2Q3 to a 16-bit system, I will assist as much as I can, but I do not have access to a "Zenith" semi- or full-compatible for programming.

Either way, if you are using Hoyle and Hoyle's QUERY!3 system, I think you will find TXT2Q3 an invaluable utility. It supplies the ability to convert data files created by other programs to this powerful, yet reasonably-priced, database package.

Vendors Mentioned

Heath/Zenith Users' Group (HUG)
PBBS:

Listing 1

```
10 REM                    TXT2Q3.MBS
20 REM  ASCII-to-QUERY!3-datafile conversion utility
30 REM                    version 1.2
40 REM                         by
50 REM                    Kirk L Thompson
60 REM                    #6 West Branch Mob Hom Vil
70 REM                    West Branch, IA
80 REM                    December, 1986
90 REM
100 REM Copyright (c) 1986 by Kirk L Thompson
110 REM This program is in the public domain and may be freely distributed!
120 REM Commercial distribution is NOT permitted!
130 REM
140 REM VARIABLE TABLE:
150 REM ================================================
160 REM A$,B$,C$Random access buffers
170 REM ABRT    Abort-program flag
180 REM ANS$    General string answer
190 REM BCT!    Byte count in destination file
200 REM BLIN    Text-file-inter-record-blank-line flag
210 REM DF$     Destination file name
220 REM DSS     Data-start sector (DOS dependent)
230 REM EF      External field-specification file flag
240 REM EF$     External field-specification filename
250 REM FALSE   Boolean equivalent (=0)
260 REM FLAB$() Field label array
270 REM FLNG()  Field length array
280 REM FTR     Truncated-field flag
290 REM I,J     FOR..NEXT loop variables
300 REM INFLD$()Input field array
310 REM MAXFDS  Max no. of fields (DOS dependent)
320 REM NFDS    Number of fields in record
330 REM NUL$    ASCII null character (00hex)
340 REM OFLD$() Output field array
350 REM PAD$    QUERY!3's pad character (80hex)
360 REM PF$     Print format string for PRINT USING
370 REM RBSIZ   Random buffer size (DOS dependent)
380 REM RCT     Read/write record counter
390 REM RDST Truncated-field¬in-destination-record counter
400 REM RLNG    Destination record length
410 REM S%      Buffer size for random access (DOS dependent)
420 REM SF$     Source file name
430 REM SSIZ    Physical sector size (DOS dependent)
440 REM           (legal values: 128, 256, 512, & 1024)
450 REM TRUE    Boolean equivalent (=-1)
460 REM
470 CLEAR 1000:WIDTH 255:DEFINT A-Z:TRUE=-1:FALSE=0:PAD$=CHR$(128):@
    DEBUG=FALSE:ABRT=FALSE
480 RBSIZ=256:SSIZ=256:IF (SSIZ MOD 128)<>0 THEN PRINT @
    "!-->ABORTING for improper value of `SSIZ' in line 480!":END:@
    REM MBASIC default random access & DOS sector size, here for HDOS
490 IF SSIZ=128 THEN MAXFDS=((SSIZ*2)-6)\17:DSS=2:@
    ELSE MAXFDS=(SSIZ-6)\17:DSS=1
500 DIM INFLD$(MAXFDS),OFLD$(MAXFDS),FLAB$(MAXFDS),FLNG(MAXFDS):@
    BLIN=FALSE:NUL$=CHR$(0)
510 PRINT:PRINT TAB(20) "T E X T - F I L E   T O   Q U E R Y ! 3":PRINT:@
    PRINT TAB(22) "D A T A F I L E   C O N V E R T E R"
520 REM
530 REM***get & check source file name***
540 REM
550 PRINT:INPUT"Enter source filename (dvd:fname.ext)--";SF$
560 PRINT:PRINT"Is --> ";SF$;:INPUT" <-- OK (Y/N)<Y>";ANS$
570 IF LEFT$(ANS$,1)="N" OR LEFT$(ANS$,1)="n" THEN 550
580 PRINT:ON ERROR GOTO 1800
590 OPEN"I",#1,SF$:CLOSE:REM check existence of source file
600 IF INSTR(SF$,".")>0 THEN EF$=LEFT$(SF$,INSTR(SF$,".")-1)+".CRE"@
    :REM set up name for field-spec file
610 IF INSTR(SF$,".")=0 THEN EF$=SF$+".CRE"
620 REM
630 REM***get & check destination file***
640 REM
650 PRINT:INPUT"Enter destination file (dvd:fname)[.DTB assumed]--";DF$
660 IF INSTR(DF$,".")>0 THEN PRINT "!-->Extension not allowed!":GOTO 650
670 DF$=DF$+".DTB":PRINT
680 PRINT"Is --> ";DF$;:INPUT" <-- OK (Y/N)<Y>";ANS$
690 IF LEFT$(ANS$,1)="N" OR LEFT$(ANS$,1)="n" THEN 650
700 IF INSTR(DF$,":")>0 THEN DVD$=LEFT$(DF$,INSTR(DF$,":"))
710 OPEN"I",#2,DF$:CLOSE:REM check existence of destination file
720 PRINT"!--> File already exists!":PRINT:GOTO 650
730 GOSUB 1870:IF EF GOTO 1000
740 REM
750 REM***manually enter field lengths & labels***
760 REM
770 PRINT:PRINT "Number of Fields (1-";RIGHT$(STR$(MAXFDS),2);
780 INPUT"):                ";NFDS
790 IF NFDS<1 OR NFDS>MAXFDS THEN PRINT "!--> Only 1 to ";STR$(MAXFDS);@
    " allowed!":GOTO 770
800 FOR I=1 TO NFDS:REM enter field lengths & names
810 PRINT "Field No. ";:IF LEN(STR$(I))=2 THEN PRINT "00";RIGHT$(STR$(I),1);:@
    ELSE PRINT "0";RIGHT$(STR$(I),2);
820 PRINT " (Length) (1-255):        ";:INPUT FLNG(I)
830 IF FLNG(I)<1 OR FLNG(I)>255 THEN PRINT "!--> Only 1 to 255 allowed!":@
    PRINT:GOTO 810
840 PRINT TAB(14) "(Name) (16 Char. Max.):  ? ";:LINE INPUT FLAB$(I)
850 IF LEN(FLAB$(I))>16 THEN PRINT "!--> 16 characters max.!":PRINT:GOTO 840
860 NEXT I
870 PRINT:PRINT TAB(20) DF$:PRINT:GOSUB 1520
880 INPUT "Save or Abort (S or A)";ANS$
890 IF LEFT$(ANS$,1)<>"S" AND LEFT$(ANS$,1)<>"s" AND LEFT$(ANS$,1)<>"A" AND @
     LEFT$(ANS$,1)<>"a" THEN PRINT "!--> No default or wrong answer!":GOTO 880
900 IF LEFT$(ANS$,1)="A" OR LEFT$(ANS$,1)="a" THEN PRINT @
     "!--> Restarting field specification!":GOTO 770
910 REM
920 REM***blank line record separator in text file***
930 REM
940 PRINT:INPUT "Is there a blank line between text-file records (Y/N)<Y>";@
     ANS$
950 IF LEFT$(ANS$,1)="N" OR LEFT$(ANS$,1)="n" THEN BLIN=FALSE ELSE BLIN=TRUE
960 PRINT:PRINT"Writing field specification file:  ";EF$;" ..."
970 OPEN"O",#1,EF$:PRINT#1,NFDS
980 FOR I=1 TO NFDS:PRINT#1,FLNG(I):PRINT#1,FLAB$(I):NEXT I:PRINT#1,BLIN:@
    CLOSE#1
990 REM
1000 REM***open input & output files; start writing***
1010 REM
1020 PRINT:PRINT "Opening files and writing destination header sector..."
1030 RCT=1:BCT!=0:ABRT=FALSE:RDST=0:FTR=FALSE
```

```
1040 OPEN"I",#1,SF$:OPEN"O",#2,DF$
1050 PRINT#2, CHR$(NFDS);CHR$((RLNG+1) MOD 256);CHR$((RLNG+1)\256);CHR$(DSS);@
     CHR$(1);NUL$;:BCT!=BCT!+6:REM first 6 bytes
1060 FOR I=1 TO NFDS:REM field lengths & labels
1070 PRINT#2, CHR$(FLNG(I));FLAB$(I);
1080 IF LEN(FLAB$(I))<16 THEN FOR J=1 TO (16-LEN(FLAB$(I))):PRINT#2,PAD$;:@
     NEXT J
1090 BCT!=BCT!+17
1100 NEXT I
1110 FOR I=(BCT!+1) TO (DSS*SSIZ):REM pad out rest of sector
1120 PRINT#2,PAD$;:BCT!=BCT!+1
1130 NEXT I
1140 REM
1150 REM***read input, process, & write output***
1160 REM
1170 PRINT:PRINT "Reading text-file record #";RCT;"..."
1180 FOR I=1 TO NFDS
1190 IF EOF(1) THEN PRINT:PRINT @
     "!--> ABORTING read for incomplete record at end of source file!":@
     ABRT=TRUE:GOTO 1350
1200 LINE INPUT#1,INFLD$(I)
1210 IF LEN(INFLD$(I))>FLNG(I) THEN OFLD$(I)=LEFT$(INFLD$(I),FLNG(I))@
     ELSE OFLD$(I)=INFLD$(I)+STRING$(FLNG(I)-LEN(INFLD$(I)),PAD$)
1220 IF NOT FTR AND LEN(INFLD$(I))>FLNG(I) THEN RDST=RDST+1:FTR=TRUE
1230 BCT!=BCT!+FLNG(I)
1240 NEXT I:FTR=FALSE
1250 IF DEBUG OR RCT=1 OR (RCT MOD 10)=0 THEN GOSUB 1690
1260 IF ABRT THEN PRINT:PRINT "!--> User-requested ABORT at record #";RCT;
     "!":@
     GOTO 1350
1270 OFLD$(1)=NUL$+OFLD$(1):BCT!=BCT!+1
1280 PRINT "Writing destination file record #";RCT;"..."
1290 FOR I=1 TO NFDS
1300 PRINT#2,OFLD$(I);
1310 NEXT I:RCT=RCT+1
1320 IF EOF(1) THEN PRINT:PRINT "!--> End of text-file!":GOTO 1350
1330 IF BLIN THEN INPUT#1,ANS$
1340 GOTO 1170:REM recycle for next record
1350 PRINT:PRINT "End of source file; closing files..."
1360 IF (FIX(BCT!) MOD SSIZ)>0 THEN @
     FOR I=(BCT!+1) TO (((FIX(BCT!)\SSIZ)+1)*SSIZ):@
     PRINT#2,PAD$;:BCT!=BCT!+1:NEXT I
1370 CLOSE
1380 PRINT:PRINT "Updating record count in destination file..."
1390 OPEN"R",#2,DF$:IF ABRT THEN RCT=RCT-1
1400 S%=RBSIZ-6
1410 FIELD#2,4 AS A$,2 AS B$,S% AS C$
1420 GET#2,1:LSET B$=CHR$((RCT+1) MOD 256)+CHR$((RCT+1)\256)
1430 PUT#2,1:CLOSE#2:PF$="##,###"
1440 REM
1450 REM***end-of-processing summary***
1460 REM
1470 PRINT:PRINT "Number of records converted:";TAB(45);USING PF$;RCT
1480 PRINT "Number of records with truncated fields:";TAB(45);USING PF$;RDST
1490 PRINT "Number of bytes written:";TAB(45);USING PF$;FIX(BCT!)
1500 PRINT "Number of physical sectors written:";TAB(45);USING PF$;@
     FIX(BCT!)\SSIZ
1510 PRINT:PRINT "File conversion complete.":END
1520 REM
1530 REM***display destination field lengths & labels***
1540 REM
1550 RLNG=0:PRINT:PRINT "FIELD NO.";TAB(15) "LENGTH";TAB(27) "NAME"
1560 PRINT "---------";TAB(15) "------";TAB(27) "----------------"
1570 FOR I=1 TO NFDS
1580 PRINT TAB(4);:IF I<10 THEN PRINT "00";RIGHT$(STR$(I),1);:@
     ELSE PRINT "0";RIGHT$(STR$(I),2);
1590 REM three-condition "case"
1600 IF FLNG(I)<10 THEN PRINT TAB(17) "00";RIGHT$(STR$(FLNG(I)),1);
1610 IF FLNG(I)>9 AND FLNG(I)<100 THEN PRINT TAB(17) "0";@
     RIGHT$(STR$(FLNG(I)),2);
1620 IF FLNG(I)>99 THEN PRINT TAB(17) RIGHT$(STR$(FLNG(I)),3);
1630 PRINT TAB(27) FLAB$(I)
1640 RLNG=RLNG+FLNG(I)
1650 NEXT I:PRINT TAB(15) "------"
1660 PRINT "Record length=";TAB(16) RLNG:PRINT
1670 RETURN
1680 REM
1690 REM***display input field labels & contents***
1700 REM
1710 ANS$="":PRINT:PRINT TAB(25) "Input record #";RCT
1720 FOR I=1 TO NFDS
1730 IF LEN(INFLD$(I))>FLNG(I) THEN PRINT "*";
1740 PRINT FLAB$(I);TAB(18)":";TAB(20) INFLD$(I)
1750 NEXT I:PRINT "*These fields will be truncated in the output file."
1760 PRINT:INPUT "Are these OK (Y/N)<Y>";ANS$
1770 IF LEFT$(ANS$,1)="N" OR LEFT$(ANS$,1)="n" THEN ABRT=TRUE
1780 RETURN
1790 REM
1800 REM***disk file error trap & handler***
1810 REM
1820 IF ERR=53 AND ERL=590 THEN PRINT "!--> File does not exist!":@
     RESUME 550:REM source file not found error
1830 IF ERR=53 AND ERL=710 THEN RESUME 730:REM ouput file not found trap
1840 IF ERR=53 AND ERL=1900 THEN PRINT"!--> File not found!":RESUME 1950:@
     REM field specification file not found trap
1850 ON ERROR GOTO 0
1860 REM
1870 REM***external field specification file check and read***
1880 REM
1890 PRINT:PRINT"Searching for field specification file..."
1900 EF=FALSE:OPEN"I",#1,EF$
1910 PRINT:PRINT"Reading:  ";EF$;" ..."
1920 INPUT#1,NFDS
1930 FOR I=1 TO NFDS:INPUT#1,FLNG(I):LINE INPUT#1,FLAB$(I):NEXT I
1940 INPUT#1,BLIN:CLOSE#1:EF=TRUE
1945 FOR I=1 TO NFDS:RLNG=RLNG+FLNG(I):NEXT I
1950 PRINT:RETURN
```

*

HTERM Modifications And Data Transfer

Ronald La Claustra
882 5th
Brooklyn, NY 11220

One of my most used HUG programs has been HTERM, an HDOS terminal emulator written by Pat Swayne in 1981. It's one of those neat and effective programs, like TEXT and PIE, that makes my H-89 still a joy to use.

With the passage of time, with changes in modem technology and remote system practices, I found it necessary to make a few small changes to HTERM. While these changes won't add autodialing or a transmission protocol (e.g., Xmodem), they do improve operations with the usual text transfers.

This will also lead to a discussion of the pitfalls of transferring text files between different operating systems.

In HTERM, the user selects, by menu, the bit rate he wants to run. Also displayed are the control sequences for opening/closing a memory storage area (buffer) to capture incoming data, and several options for transmitting data from disk files.

What's Involved

To make the changes, you'll need to alter the HTERM source code which is on the HUG disk (#885-1089) as HTERM.ASM. You will then have to use the assembler to produce the new version of HTERM.ABS that you can run. I'll try to be explicit enough for those who have never attempted this before. The descriptions of the code operation will be terse, since it's not my purpose here to teach Assembly language.

Make up a disk with the ASM.ABS program copied from your HDOS masters. Bring over your text editor and the HTERM.ASM file. These don't have to be on the same disk if you have more than one drive.

Run your editor and load it with HTERM .ASM, then make the text changes as described below. Save the new file as HTERMREV.ASM and exit your editor. Invoke the assembler by typing "ASM", then after it has loaded, type "HTERM=HTERMREV". (If you only have one drive, you may have to delete some files to give the assembler enough room to work.) After a minute or so of processing, the assembler will display some lines, one of which should say "NO ERRORS DETECTED". The new HTERM will be on the disk.

I will assume your version of HTERM is dated 20-AUG-81 at the top of the listing. As far as I know, there have been no revisions since then, although there were a couple of patches published in REMark which I will go into below. If it bears a different date, you can probably still make the changes as long as the section of code matches the given examples.

A printout of the HTERM.ASM file will be helpful as a location guide. I'll refer to the approximate location of the changes as being on the n-th page of this listing. Be sure the original lines match before you change anything and keep to the columnar format of the program.

Modem Control

The new smart modems seem to need two more of the RS-232 status lines (DTR and RTS) activated in order to work. While the modems usually have switches to force these lines if they haven't been by the computer, it's nice to run along with the rest of the (IBM?) world. By adding three lines of code, we can have HTERM set the modem up properly.

The first line goes on the first page of the listing. Listing 1 shows two lines of the original listing: the new line, then two more lines of the original code. I'll use this method of showing the changes to save space.

Note: My listings appear in the order they appear in the program not by their listing number. The exception is listing 7.

The last 2 lines of this change are shown in Listing 2. Look in the area just above the word "DELAY" in the first (left-most) column on page 3. The left-most column is used to label the blocks of code, I'll refer to these labels so you can locate the right areas.

The "EQU" line gives the program the location address of the control register for the two modem control lines. The second and third lines load the correct value to turn on the lines into the serial IC at the given address.

Bit Rates

The HTERM menu features most of the common bit (baud) rates in use, but before I upgraded to a 1200 BPS modem, I would occasionally want to run at 450 BPS which my 300 BPS modem could manage. Some modems were rated at this speed; some, like mine, could run it anyway, although fewer remote sites can recognize the speed these days. Even so, the option is nice to have even if it's just for use on private lines.

I had no use for the 134.5 rate in the HTERM menu (page 2), so I changed it to

Listing 1

```
XRPORT  EQU     330Q            TRANSMIT-RECEIVE PORT
XSPORT  EQU     XRPORT+5        XR STATUS PORT
MCPORT  EQU     XRPORT+4        MC CONTROL PORT         ADDED 9/21/86 R.L.
XLPORT  EQU     XRPORT+3        XR LINE CONTROL PORT
XIPORT  EQU     XRPORT+1        XR INTERRUPT CONTROL PORT
```

Listing 2

```
        MVI     A,3             SET UP WORD LENGTH, ETC.(7 FOR 2 STOP BITS R.L.
        OUT     XLPORT
        MVI     A,3              VALUE FOR SETTING DTR AND RTS R.L.
        OUT     MCPORT           SET UP          CHANGED 9/21/86 R.L.
        MVI     A,ION
        OUT     XIPORT           ENABLE RECEIVE INTERRUPT
        LXI     H,65000A         SET UP A DELAY
DELAY   DCR     L
```

read "450.." (Listing 3). Also needed is a change in the "BAUD RATE TABLE" at the end of the program (Listing 4). This gives the serial IC the proper divisor value.

Message

I use HTERM to upload progams to a college system a great deal. Some of these things can take quite a while, especially at 300 BPS, so I do other things while the transfer goes on. I found it helpful to have a definite notice that the transfer had come to an end, so I added one.

At the penultimate page, in the section under "DLOAD4" in the first column, insert the new lines given in Listing 5. These lines define (DB) the ending message (ring the terminal "bell" and type the message in reverse video) and call the operating sytem to print the message on screen ($TYTPTX) while saving the previous state of the program (PUSH-POP).

You may see a couple of junk characters on the screen after the message appears (buffer leftovers?), but they do not mess up the transmission. Due to other delays when using the "with echo" (Control-^) method, the last line(s) of the transfer won't appear until after the message, but it went ok.

Terminal Width On Exit

On exiting, HTERM left the terminal at a 255 column width. I preferred an 80 column width, as that's what I have HDOS set to. On the last page, in the "EXIT" section, Listing 6 shows the needed changes. The new lines load the proper values, then call the operating system, "SCALL", to do the terminal setting before ending the program.

Obviously, if you like the 255 column width, don't make this change.

New Lines

As background for the next two changes, a little history is in order . . .

When a typist came to the right end of the paper on a manual typewriter (you do remember those, don't you?), the bell rang and the typist would move a lever on the left of the paper carriage across to the right. This action brought back the left edge of the paper and simultaneously pulled a ratchet which rolled the paper up one line for a new line of typing.

When teletype machines were invented, the definers of the electrical codes (ASCII), which would control the machines, kept this two stage process by having one code return the carriage (or print head) to the left edge, and a second code advance the paper upwards. In the ASCII code, these codes are called "carriage return", character number 13 (decimal), and "line feed", character 10. With the advent of electronic display terminals, two codes were no longer needed to trip motor controlling solenoids. So some software writers dispensed with one of the codes to save space and time, then used the remaining code as their "new line" indicator.

The conflicts arose when text prepared on a system that used the carriage return only, was sent over to a system which used the line feed only. The opposite case is just as bad, and the case where one system retains both characters causes more of a mess.

The results of these incompatibilities depends on which system wants what, and what the receiving system does when it gets the unused control character. An unending line or one line which is continuously overwritten is what this looks like.

HDOS was developed using the line feed (LF) as its new line code. The old TRS-80 systems used the carriage return (CR). As a result, the old remote systems using a TRS computer would always ask if you needed to be sent the line feed when you signed on, otherwise, it wouldn't bother.

The newer MS-DOS systems, as well as the CP/M systems, use both CR and LF to move to a new line. Thus, an HDOS user needs to keep these points in mind, and also have the software adapt to these things.

The Patches

HTERM behaves differently with the two cases of normal data and saved data. From the keyboard going out, HTERM sends out whatever each key is, so the "RETURN" key sends out a CR and the "LINE FEED" sends out (surprise!) an LF. But when data is sent from, or is to be stored on disk, HTERM makes changes. Going from the disk out, HTERM changes the HDOS LFs to CRs. Incoming control codes are filtered.

Originally, HTERM didn't store any control code, but LF. In REMark #'s 29 and 30, Pat published a couple of patch changes to the program. One fix was a spelling correction of the word "Operation" in "Operation aborted" in a line just above the block of text describing "SBUFF" (6th page).

Listing 3

```
DB      'Set Baud Rate to 110.............1'
DB      ESC,'Y',31+5,31+24
DB      'Set Baud Rate to 450.............2'  CHANGED R.L. 10/12/84
DB      ESC,'Y',31+6,31+24
DB      'Set Baud Rate to 150.............3'
```

The second fix allowed the storage of all the ASCII characters, except the CR. Listing 7 shows the original block of code. It's in the "SBUFF" section just below the spelling fix. Listing 8 shows what it should be changed to. This change allows all the other control characters in, so you can save tabs or escape codes in the buffer. And it still keeps any transmitted CRs from being troublesome when you later use your captured text. I've used HTERM this way for years with no trouble from "junk" codes creeping in from the transmitting site.

Note: Pat used octal notation to give the values for CR & LF so they appear as 12Q (the LF) and 15Q (CR) in the listings.

'Plex And Disk Transmission

I mentioned I upload to a college system. It's surprisingly old fashioned in that it only

Listing 4

```
BTABLE  EQU     *               BAUD RATE TABLE
        DW      027004A         110 BAUD
        DW      000001A         450 BAUD        CHANGED HERE R.L. 10/12/84
        DW      000003A         150 BAUD
        DW      200001A         300 BAUD
```

Listing 5

```
CALL    $CDEHL          END OF FILE?
JNZ     DLOAD2          LOOP UNTIL END
PUSH    PSW                                     ADDED HERE DOWN TO
CALL    $TYPTX
DB      10,7,'                          ',27,112,0,0,'END OF TRANSMISSION'
DB      27,113,0,0,0,0,0,0,10
POP     PSW                                     HERE 11/11/84
MVI     A,ION
OUT     XIPORT          ENSURE MODEM INT. ON
```

works in half duplex. This is where your computer sends out a character and the receiver does nothing back. Newer systems (all the BBS I ever heard of) get the character from you, then send what they got back to you (echo) when it then appears on your screen. This roundabout route insures accurate reception, since if you type a "g" and get back a "P", you know there has been an error. Since HTERM expects this full duplex, what I saw was a static cursor when I typed and normal printing when the college wrote. Fortunately, I could set my modem to half duplex so that it echoed my typing, but there remained a problem in the transmission of files from my disk. (More on half duplex below.)

Listing 6

```
MVI     A,IOFF          TURN OFF
OUT     XIPORT          MODEM PORT INTERRUPTS
MVI     A,3                     ADDED HERE
LXI     B,050FFH                                !
SCALL   .CONSL          SET CONSOLE WIDTH TO 80 COL. TO HERE RL 11/11/84
XRA     A
SCALL   .EXIT           EXIT TO HDOS
```

HTERM has several possible methods of transmitting from your disk. The first just spews the characters out and hopes the receiver can catch them. The college couldn't cope with this one. (Neither could any other system: it's generally useless.)

The second is more secure from errors in that it waits for the echo before sending the next character, but if it doesn't get the echo, the program will just sit there until you get tired of waiting and abort it. Obviously, I'd have a long wait if I used this on the college system. (Note that HTERM doesn't check to see if the echo is the same character it sent out, anything will do. If the remote system does something like add line numbers, this may throw off the transfer.)

Listing 7

```
        CPI     12Q             NEW LINE CHARACTER?
        JZ      SBUFF1          IF SO, STORE IT
        CPI     40Q             OTHER CONTROL CHARACTERS?
        RC                      DON'T STORE THEM
SBUFF1  MOV     M,A             STORE CHARACTER
        INX     H               BUMP POINTER
```

Listing 8

```
        CPI     12Q             NEW LINE CHARACTER?
        JZ      SBUFF1          IF SO, STORE IT
        CPI     15Q             CARRAGE RETURN CODE?        PATCH CHANGE
        RZ                      DON'T STORE IT              2/3/83 R.L.
SBUFF1  MOV     M,A             STORE CHARACTER
        INX     H               BUMP POINTER
```

The third method is the potential winner. In addition to waiting for an echo as above, it checks the characters as they go out, and if you are sending a "new line" character (LF in the disk file), HTERM waits a period of time before sending the next character. At least that was supposed to happen. The actual delay seemed to be on the order of milliseconds when I tried to use it. It was only when I analyzed the source code that I found the reason. There seems to be a problem in the part of HTERM that deals with this new line delay. I'm not certain of what was intended or the type of remote system HTERM was designed for, so I won't call it a bug.

Listing 9

```
        CPI     12Q             NEW LINE?               CHANGED TO SEND CR/LF
        JNZ     DLOAD3          IF NOT, CONTINUE        END OF LINE
        MVI     A,15Q           REPLACE WITH CR             HERE TO
        CALL    TRAN            TRANSMIT THE CR
        LDA     CONCHR
        CPI     CTLAT           FULL SPEED MODE?
        JNZ     SENDLF          NO, SKIP
      MVI     A,15Q
        CALL    OUTCH           YES, PRINT CARRAGE RETURN   HERE R.L. 2/14/87
SENDLF  MVI     A,12Q           RELOAD LF
DLOAD3  CALL    TRAN            TRANSMIT THE CHARACTER
        STA     LASTCH          STORE LAST CHARACTER
```

As noted above, HTERM switches LFs to CRs when it transmits from a disk file. However, on the echo, HTERM looked for an LF when it sent out a CR, thus the delay was never invoked.

Line End Compatibility

To fix this and make HTERM more compatible with the rest of the world, I replaced the non-working code (begins under DLOAD2, page 8) with the block in Listing 9. The new code detects an LF from the disk file and replaces it with a CR/LF pair. This works with all BBS systems I've tried.

Because the high speed transfer choice (Control-@) is handled differently, the new code also prints a CR to the screen so the transmitting data displays correctly on your screen.

With this change, the three types of transfers all work, and display identically on your screen. I found, however, that the "new line" delay was too short, thus Listing 10 (page 9, XWAIT block down) shows how I added extra "do nothing" loops to the program to increase the delay to about 1.5 seconds on a 2MHz H-89. The delay will be about half this on a 4MHz CPU clock.

ECHO

Now I ran into some problems with my odd, old college system. My old modem had a half duplex switch which, as explained above, allowed me to see my own typing. But my new high speed modem doesn't have such a capability, so I had to change HTERM again. On page 5 at the end of the TRAN0 block, add the line: CALL (2nd col.) OUTCH (3rd col.) just over the final line RET. This has HTERM print what you type as it's being sent out.

There remains an insoluable problem though, I couldn't transmit files from the disk because there is no echo from the outside: the program will hang up. The moral is be certain the modem you want to buy will do what you need. (As another example, some 1200 bps modems can't pass a "BREAK" which my college system needs.)

CR Only?

The college system also didn't like the revised file transmission in HTERM. It balked at having the CR precede the LF (what else would you expect from an IBM behemoth?). If you should be shackled to such a system, instead of using Listing 9 to change the from-disk code, just change the original value of 12Q in line WAICHR1 to 15Q. This will have HTERM recognize an echoed CR as the character to put the delay after.

Data Formats

That's it for the modifications. If your working on a system that needs some odd method of character transmission, use the values in the following table. Then change the value in the line of Listing 2 located just above the line where you made the DTR/RTS addition, and also in the BREAK section (page 5). The line reads: MVI A,3 followed by a line which reads: OUT XLPORT in both cases. It's the 3 that gets changed.

Listing 10

```
        MOV     A,B             RESTORE CHARACTER
        RNZ                     RETURN IF NO DELAY
* MODIFIED TIMING LOOP HERE DOWN R.L. 10/1/84
XWAIT2  LXI     B,0FFFFH        LOAD B,C MIDDLE LOOP COUNT
EZIP    PUSH    B               SAVE MIDDLE
        LXI     B,0FFFFH        RELOAD B,C FOR WAIST TIME LOOP
ZIP     DCR     C
        NOP
        JNZ     ZIP             KILL TIME
ZIP1    DCR     B
        NOP
        JNZ     ZIP1
XWAIT1  POP     B               GET MID LOOP B,C BACK
        DCR     C               DEC AND
        JNZ     EZIP                    CONT IF NOT ZERO
        MVI     C,1             KEEP C FROM GOING AROUND ZERO
        DCR     B               DEC AND
        JNZ     EZIP                    CONT IF NOT ZERO, ELSE
        RET

* ERROR HANDLER
```

To find your desired value, find the parameter in each vertical column and add the given values horizontally. Thus, 8 bits, one stop bit, no parity equals 3. 7 bits, 2 stop bits, odd stick parity equals 46.

5 bits 0 1 stop bit 0 no parity 0
6 bits 1 2 stop bits 4 even parity 24
7 bits 2 odd parity 8
8 bits 3 odd stick parity 40
even stick parity 56

Make notes as to the changes you made to HTERM at the top of the progam under the description. Precede each line with an asterisk (*).

I hope this will help you get the most out of a very useful HUG program, and solve your text transfer troubles. *

C/80 In OMDOS, HDOS 3.0 [and M-80]

Don Keller
1330 Eden Valley Road
Port Angeles, WA 98362

C/80 remains the most versatile and useful languages available for 8-bit microcomputers, but I have to admit it's a little short on elbow room in my single drive H-89. Due to Heath's early release of operating system source code, though, a lot of talented people have been able to improve the situation considerably over the last few years.

OMDOS

One Module Disk Operating System (OMDOS) developed by Skip Chambers is a fine example. Like HDOS 3.0, (which we'll get into later) OMDOS eliminates the use of overlays and their associated problems. There is enough room in memory to run the C compiler even with a large device driver resident. I use UD.DVD to run my printer. It's a powerful driver, but it occupies four kilobytes of RAM. PIP and ONECOPY are incorporated into SYSCMD .SYS and always available. OMDOS is always in stand alone mode. The H-17 driver that comes with OMDOS is quiet and supports double-sided drives. I've been using OMDOS with C/80 and MATHPAK, PIE, TEXT and SPELL for more than a year with good results. See the appendix to this article for Skip's address.

HDOS 3.0

It was a long wait, with plenty of opportunities for those of little faith to give up hope, but HDOS 3.0 is now in circulation. You can download it from the HUGPBBS if you have the rather extensive collection of equipment required. I got mine by mail, directly from Bill Parrott. (See the appendix.) HDOS 3.0 is now in the public domain, so you can also get a copy from anybody who has it — pass it on! It looks pretty good; it also looks pretty complicated. You can be entertained for a LONG time just finding out what it can do. Disk types that are supported include 8", 5-1/4" (hard- or soft-sector, single- or double-sided, 48 or 96 tracks per inch), and the Bernoulli Box for crying out loud. That last one may be more gee whiz than anything else; the super density 8" floppies for a Bernoulli Box have to be formatted by a program that runs in MSDOS. The list of utilities in SYSCMD and PIP is huge. Fortunately, very good HELP files are included.

UD.DVD, Softshop's Universal Device Driver, will not work in HDOS 3.0. I don't know why not; I replaced all the .ACM files with new ones for version 3 and it assembles without error. All the options are settable. Bill Parrott's README.DOC file says the old style interrupt implementation in device drivers will crash HDOS 3.0, but UD.DVD doesn't cause a crash. It does nothing at all and you can get out with a CTRL-C. I've written to Jim Teixeira of Softshop begging for help, but at the time of this writing I haven't heard anything. On the other hand, many printers are supported by drivers delivered with HDOS 3.0 and you can load or unload drivers whenever you like.

MACRO-80

On the back of my MACRO-80 manual there is a notice that says, in part, "Your computer must have dual-drive floppy disk capacity..." HOGWASH! MACRO-80 and LINK-80 make it possible to program in C/80 on a single drive system without worrying about the size of the programs. Well — within reason. No doubt you can overflow any system, but if you've spent a year or two trying to hoard room for C programs without a relocating assembler, you'll think you're suddenly working without restrictions on disk and RAM space. Trouble is, you could have a hard time getting MACRO-80. You might try Schumacher Associates. You might even be able to get through to Microsoft. (Both in the appendix.) I'm treading on thin ice to say this, but a software vendor shouldn't complain about piracy of a product they refuse to support any longer.

The System

What do you do with two kinds of HDOS beyond version 2.0? I've decided I like OMDOS better for word processing. It allows the use of UD.DVD, and its SY.DVD operates the drive with less fuss and noise. For programming in C/80, HDOS 3.0 is the champ. The facilities provided by SYSCMD and PIP are hard to beat: file manipulation, including setting and clearing flags; multiple commands on a single line, plus batch file processing; a software real time clock background task with time stamping of files in the directory; and the list goes on. So

let's expand a little on the use of HDOS 3.0 for C/80.

The standard ONECOPY is too big to put on every disk. I have it on the system disk which spends most of its time in a box. A RESET0 permits remounting the system disk whenever ONECOPY is needed. Most disk-to-disk transfers during programming, though, are one file at a time. QCOPY in Listing 3 is a utility that takes one file into memory, resets SY0:, and writes the file to the disk you put in the drive. QCOPY exits with the new disk mounted. This is convenient to move a C source file from the editor disk to the compiler disk, a .MAC or .ASM file to the assembler disk, and perhaps a .REL file to the linker disk. Each of these disks can have its own QCOPY.ABS; it's only 10 sectors long.

HOSLIB.C

The first time I browsed through STDLIB.C distributed with C/80, I was annoyed to find that two important functions were marked, "CP/M only": rename() to rename disk files from within a C program, and unlink() to delete a disk file. They were probably left out of the HDOS version because of possible memory use conflict caused by system overlays. Both rename() and unlink() need to use overlaid SCALLs in HDOS, but neither OMDOS nor HDOS 3.0 uses overlays, so we ought to have those functions. Listing 2, HOSLIB.C, provides those two and a few more besides. See the Operating Manual section below.

If you use MACRO-80 with HDOS 3.0, you may have some trouble with error conditions. When LINK-80 fails to find a global called by your program, it tries to load FORLIB.REL (the FORTRAN library) to search for it, WHETHER YOU HAVE FORLIB.REL OR NOT. Under HDOS 2.0 and OMDOS it just says it couldn't find FORLIB.REL, but HDOS 3.0 crashes. The solution is to give LINK-80 a FORLIB.REL. Listing 4 shows a version that solves this problem and one other at the same time.

The release of MACRO-80 I have is an early one with no library manager. Typing a long list of library .REL files every time I ran the linker got old in a hurry, so I wrote a little routine that searches library files. When I discovered the FORLIB crash in HDOS 3.0, I named this file FORLIB.REL and took care of that problem, too. If you're one of the expansive types with plenty of mass storage AND have a real FORLIB.REL in residence, you'll obviously have to name this one something else: CLIB.REL, for instance.

Operating Manual

Listing 1 — stdio.h

This is the header file that contains definitions frequently needed in C programs. Naturally, you can add any definitions you want or delete any you never use. You may have to increase the size of C/80's #define table if you define very many more labels. See section 6.2 in your C/80 manual and run CCONFIG.ABS (or CCONFIGF.ABS for the MATHPAK implementation).

SCALL

MACRO-80 does not recognize the HDOS SCALL. This first piece of stdio.h defines SCALL as a macro so that Assembly language routines included in C programs can use SCALL when you use MACRO-80. When AS.ABS is used instead of MACRO-80, this segment should be deleted or commented out or you'll get complaints from the assembler.

The next block of text is taken directly from the C/80 manual. These are redefinitions to suit current style. There is more involved than a satisfactory looking source file; a standard style tends to ease portability to and from other systems. fin and fout must be declared extern since they are declared in a C/80 file. stderr, the standard error output is defined as (channel) 0 — always the terminal. stdin is the standard input, stdout is the standard output; both are the terminal UNLESS redirected by the command line or reassigned by your program.

ESC, BELL, EOF, CR, and NL are numeric constants commonly needed.

NULL, YES, NO, ON, OFF, TRUE, and FALSE are handy for flag values, etc.

INTSIZ is the length of a character string big enough to hold the ASCII representation of a 16-bit integer, plus a terminating NULL byte.

NAMSIZ is the length of a character string big enough to hold the ASCII representation of an unambiguous file name, plus a terminating NULL byte.

TTLINE is the length of a console line, plus a terminating NULL byte.

LINE is long enough for virtually any input line.

C_MODE is the argument for the .CONSL SCALL or the conmod() function to set the system console to character mode without echo.

L_MODE is the argument for the .CONSL SCALL or the conmod() function to set the system console to line mode with echo (normal).

The remainder of stdio.h is a selection of SCALL's. To make them acceptable to C/80, AS.ABS and MACRO-80, the dot character has been replaced with the underline character; their length has been limited to six characters; and numeric values are decimal. The compiler and both assemblers accept other number bases, of course, but each uses a different syntax to identify bases other than ten.

Listing 1

```
/*      listing 1
 *
 *      stdio.h          for HDOS, C/80 and M80
 */

#asm
SCALL     MACRO   CALL_ID          /* define SCALL as a macro. */
        DB      255,CALL_ID      /*      (delete or comment  */
        ENDM                     /*      out for use with    */
#endasm                          /*      AS.ABS)             */

#define FILE    int      /* from             */
extern int fin, fout;    /*                  */
#define stdin   fin      /*      C/80        */
#define stdout  fout     /*                  */
#define stderr  0        /*           manual */

#define ESC     27       /* ESCape */
#define BELL    7        /* console alarm */
#define EOF     -1       /* End Of File */
#define CR      13       /* carriage return */
#define NL      10       /* newline (line feed) */
```

```
#define NULL    0
#define YES     1
#define NO      0
#define ON      1
#define OFF     0
#define TRUE    1
#define FALSE   0

#define INTSIZ  7       /* int string size */
#define NAMSIZ  17      /* unambiguous file name size + 1 */
#define TTLINE  81      /* console line length + 1 */
#define LINE    128     /* room for overlength line */

                        /*      for SCALL _CONSL or conmod() */
#define C_MODE  33153   /* character mode, without echo      */
#define L_MODE  129     /* line mode, with echo (normal)     */

#define _SCIN   1       /* HDOS system calls (SCALL's) */
#define _SCOUT  2
#define _CONSL  6
#define _CLRCO  7
#define _VERS   9
#define _CLOSE  38
#define _DELET  40
#define _RENAM  41
#define _ERROR  47
#define _RESET  132

 __DELET   40
#define __RENAM   41
#define __ERROR   47
```

Listing 2 — hoslib.c

#include <stdio.h> — requires definitions from Listing 1 above.

chgsy0() — This is primarily intended to make life easier for single drivers, but it may be useful on larger systems. It allows a C program to prompt for and execute a disk change in SY0:. Close all open files before you call chgsy0 or be prepared for interesting results. Usage:

```
chgsy0();
```

clrco() — Clears the HDOS type ahead buffer. Sometimes an input function leaves characters in the buffer when your program exits. Then HDOS thinks it's a mistyped command and says so. Using clrco just before an exit will keep you from seeing one more error message. Usage:

```
clrco();
```

conmod(mode) — Sets or reports the system console mode. Conmod returns an unsigned 16 bit value. Usage:

```
unsigned rm;

rm = conmod(NULL);
    /* mode unchanged, rm contains   */
  /* current mode (C_MODE or L_MODE) */
conmod(C_MODE);
    /* character mode without echo */
conmod(L_MODE);
    /* line mode with echo (normal) */
```

C_MODE, L_MODE and NULL — Defined in stdio.h, Listing 1.

*** conmod works with getc and getchar BUT NOT with scanf!!! ***

(Single character input with scanf will always run one keystroke behind.)

fgets(buffer, count, chan) — Reads a character string from a file or device. 'buffer' is a pointer to (or array name of) character storage declared by the caller to fgets. A maximum of 'count' minus 1 characters will be read. (The minus 1 leaves room for a terminating NULL byte which is added to the string by fgets.) 'chan' is the number of a channel opened for read. It could also be the standard input, 'stdin,' which doesn't have to be explicitly opened or closed. There are many versions of fgets; this one returns EOF when it encounters End Of File, NULL if it encounters more characters than it asked for, or the number of characters read which indicates the string is stored in memory pointed to by 'buffer'. DO NOT use fgets and scanf in the same program. For that matter, you can't use getc, getchar, or getline in a program with scanf, either. scanf uses an ungetc function to push unused characters "back onto the input channel" and none of the other input routines know about it. Usage:

```
char str[100];
     [int     n;]
   /* text in [brackets] is optional */
      FILE    inch;

      [n = ]fgets(str, 100, inch);
```

fputs(s, chan) — There are also plenty of versions of fputs around. This one writes a string pointed to by 's' onto channel number 'chan' which must be opened for write or update. 'chan' can also be the standard output, 'stdout' or the standard error output, 'stderr,' neither of which has to be explicitly opened. This version of fputs DOES NOT add a new line to the string being written. Usage:

```
static char str[] = "Hi, there!\n";

fputs(str, stderr);
```

or:

```
fputs("Hi, there!\n", stderr);
```

rename(s, t) — Renames a file. 's' points to a string containing the old name and 't' points to a string containing the new name. rename returns -1 if an error occurs and 1 for success. This HDOS version was designed to behave just like the CP/M version supplied with C/80. Usage:

```
 static char old[] = "foo.bar";
 static char new[] = "sna.foo";
[int n;]

 [n = ]rename(old, new);
```

or:

```
rename("foo.bar", "sna.foo");
```

unlink(s) — Deletes (unlinks) a file. 's' points to a string containing the name of the file to be deleted. This too is made to look like the CP/M version. No useful value is returned. Usage:

```
static char trash[] = "sna.foo";

unlink(trash);
```

or:

```
unlink("sna.foo");
```

HOSLIB.C contains two global identifiers, rename and unlink, which are duplicates of names found in STDLIB.C. Customary usage of STDLIB.C when AS.ABS is to be used is to have C/80 selectively search STDLIB.C for functions called by your program by placing #include\ <STDLIB.C> at the end of your source code. If you say:

```
#include <hoslib.c>
#include <stdlib.c>
```

IN THAT ORDER, C/80 will compile rename() or unlink() — if they're called by your program — from HOSLIB.C and ignore them when it finds them in STDLIB.C, since the search will already be satisfied. The situation is much the same when MACRO-80 is used; as long as HOSLIB.REL is linked BEFORE STDLIB.REL, rename and unlink will be ignored in STDLIB.REL. You can save some disk space, though, by editing those functions out of STDLIB.C. You can also delete bdos() and makfcb() which are CP/M crutches.

Listing 2

```
/*              listing 2
 *
 *      hoslib.c        additions to STDLIB.C, C/80
 */

#include <stdio.h>

chgsy0()        /* reset SY0: to change disks */
{
#asm
        MVI     A,-1    /* HDOS channel that loaded program */
        SCALL   _CLOSE  /* close it to avoid open file error on _RESET */
        LXI     H,DRSPEC   /* HL points to drive specification string */
        SCALL   _RESET  /* do the system call */
        LXI     H,1     /* return 1          */
        RNC             /*      if no error */
        LXI     H,BELL  /* trailing character for _ERROR */
        SCALL   _ERROR  /* use ERRORMSG.SYS to report error */
        LXI     H,NULL  /* return NULL if error */
#endasm
}

#asm
DRSPEC: DB      'SY:',0         /* drive specification for _RESET */
#endasm

clrco()         /* clear HDOS typeahead buffer */
{
#asm
        SCALL   _CLRCO
#endasm
}

unsigned conmod(mode)   /* set mode for keyboard input */
{               /* 'mode'  (defined in stdio.h)                         */
#asm                    /*      C_MODE -> character mode, no echo       */
        POP     H       /*      L_MODE -> line mode with echo           */
        POP     B       /*      NULL ->   change nothing, return mode */
        PUSH    B
        PUSH    H
        XRA     A
        SCALL   _CONSL
        LXI     H,L_MODE
        MOV     H,A     /* return new mode value */
        RNC             /*     if no error       */
        LXI     H,BELL  /* else use                        */
        SCALL   _ERROR  /*     HDOS error message and      */
        LXI     H,NULL  /*          return NULL to caller */
#endasm
}

/***** DO NOT use fgets and scanf in the same program!!! *****/

fgets(buffer, count, chan)      /* get string from file or device */
char *buffer;
int count, chan;
{
        int c, n;

        n = 0;
        while (n++ < count && (c = getc(chan)) != '\n' && c != EOF)
                *buffer++ = c;
        if (c == EOF) return EOF;
        *buffer = '\0';
        if (--n < count) return n;
        else return NULL;
}

fputs(s, chan)  /* write string to file or device */
char *s;
int chan;
{
        while (*s) putc(*s++, chan);
}

rename(s,t)                     /* rename a file from 's' to 't' */
{
#asm
        POP     D       /* function return address */
        POP     B       /* BC points to new name string */
        POP     H       /* HL points to old name string */
        PUSH    H       /* restore -       */
        PUSH    B       /*       ---       */
        PUSH    D       /*         - stack */
        LXI     D,DEFALT   /* DE points to default string */
        SCALL   _RENAM  /* do the system call */
        LXI     H,1     /* return 1          */
        RNC             /*      if no error */
        LXI     H,-1    /* else return -1   */
#endasm
}

unlink(s)                       /* delete file named 's' */
{
#asm
        POP     B       /* function return address */
        POP     H       /* HL points to name string */
        PUSH    H       /* restore    */
        PUSH    B       /*      stack */
        LXI     D,DEFALT   /* DE points to default string */
        SCALL   _DELET  /* do the system call */
#endasm
}

#asm
DEFALT: DB      'SY0',0,0,0     /* default string for _RENAM and _DELET */
#endasm
```

Listing 3 — qcopy.c

#include <stdio.h> — needs header file in Listing 1 above. Also, calls Routines in Listing 2 above.

qcopy is a small (10 sectors in QCOPY.ABS) utility to read one file from a disk, reset SY0:, write the file to the new disk and exit with the new disk mounted. Usage:

```
(at the system prompt) qcopy filename
   <RET>
```

If you fail to supply a file name, qcopy will remind you to include one on the command line, and exit to HDOS. Start over.

qcopy reads the file and HDOS prompts for a disk change. Put the destination disk in SY0:

qcopy writes the file on the new disk and exits with the new disk mounted. Any error conditions during the process are reported by either qcopy or HDOS.

Listing 4

CLIB.C — A sample of how to make life bearable in the absence of a library manager for LINK-80. CLIB.C is heavily commented and pretty self-explanatory. CLIB should be compiled with C/80 and assembled with MACRO-80. IF you don't have FORLIB.REL on the linker disk, rename CLIB.REL to FORLIB.REL. Otherwise, HDOS 3.0 will no doubt crash whenever LINK-80 fails to find a global referenced by your program. Usage:

```
(at the system prompt) L80 prog,
   forlib,prog/n/e <RET>
```

or:

```
L80 prog,clib,prog/n/e <RET>
```

Listing 3

```
/*      listing 3
 *
 *      qcopy.c         special onecopy for c/80 environment
 */

#include <stdio.h>

#define BUFSIZ  32768   /* 32 K buffer */
#define U_CHAR  256
#define S_CHAR  128

main(argc, argv)
int argc;
char *argv[];
{
        FILE inch, ouch;
        char *bufin, *bufout;
        int d, i;

        if (argc != 2) {
                fputs("name file in command line\n", stderr);
                exit();
        }
        if ((inch = fopen(argv[1], "rb")) == NULL) {
                fputs("can't open ", stderr);
                fputs(argv[1], stderr);
                putc('\n', stderr);
                exit();
        }
        bufin = bufout = sbrk(BUFSIZ);
        for (i = NULL; i < BUFSIZ; ++i) {
                if ((d = getc(inch)) == EOF) break;
                if (d >= S_CHAR) d -= U_CHAR;
                *bufin++ = d;
        }
        if (i == BUFSIZ) {
                fputs(" too big for QCOPY.\n", stderr);
                exit();
        }
        *bufin = EOF;
        fclose(inch);
        chgsy0();
        if ((ouch = fopen(argv[1], "wb")) == NULL) {
                fputs("can't make ", stderr);
                fputs(argv[1], stderr);
                putc('\n', stderr);
                exit();
        }
        while (i--) {
                d = *bufout++;
                if (d < NULL) d += U_CHAR;
                putc(d, ouch);
        }
        fclose(ouch);
}
```

And In Conclusion

When the functions and utilities just described are added to your collection of C tools, and combined with the power and convenience of MACRO-80 and either OMDOS or HDOS 3.0, you won't have to defer to anybody's software development system. That's the beauty of C; it doesn't much care whether your computer is a super mainframe or an H-89. er mainframe or an H-89.

Appendix

OMDOS (Inquire)
Skip Chambers
3822 Westminster Drive
Carrollton, Texas 75007

MACRO-80 (Inquire)
Microsoft, Inc.
16011 NE 36th Way
Redmond, WA 98073-9717

HDOS 3.0 $25.00
(Specify H-17, H-37, or H-47)
Bill Parrott
7010 Caenen Avenue
Shawnee, KS 66216

MACRO-80 (Inquire)
Schumacher Associates
12619 Valleywood Drive
Woodbridge, VA 22192

UD.DVD (Inquire)
Softshop
35 Shadow Oak Drive
Sudbury, MA 01776

C/80, MATHPAK, PIE, TEXT, SPELL, Etc.
(Ask for a Catalog)
The Software Toolworks
14478 Glorietta Drive
Sherman Oaks, CA 91423

Listing 4

```
/*      listing 4
 *
 * clib.c       direct L80 to search .REL files for C/80 library
 *
 * The .REQUEST statement causes L80 to search the named files for
 * undefined globals.  This enables the inclusion of library routines
 * with a single file request even in the absence of LIB80.
 *
 * With clib compiled, assembled and present on the L80 disk as CLIB.REL,
 * a command line like
 *      >l80 prog,clib,prog/n/e
 * will search all the .REL files .REQUESTed in clib for any routines called
 * by prog.  The /s switch is not used; it's implicit in .REQUEST.
 *
 * Order of appearence in the code below may be critical; if anything in
 * in one .REL file calls a routine in another, the caller MUST appear in
 * the list ahead of the file containing the called routine.  With that
 * restriction in mind, you can add or remove any .REL files you like.
 *
 * M80 ignores characters in excess of six in labels, so some C/80 files
 * have to be renamed.  These are identified in the code below by a comment
 * showing the original name.  All files .REQUESTed must be present on the
 * L80 disk.
 */
#asm
        .REQUEST        fprint          /* fprintf.rel */
        .REQUEST        scanf
        .REQUEST        seek
        .REQUEST        exec
        .REQUEST        commnd          /* command.rel */
        .REQUEST        hoslib
        .REQUEST        stdlib
        .REQUEST        mathlb          /* mathlib.rel */
        .REQUEST        flibry          /* flibrary.rel */
        .REQUEST        clibry          /* clibrary.rel */
#endasm
```

*

PAINT.ASM Again?

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

An Alternative Printer Routine To Permit A Wider Selection Of Printers To Use PAINT.ASM!!

The Problem

WOW! This past year has given new meaning to time flying. Well, I haven't been idle either. From the letters, it has been apparent that the weakest portion of the "PAINT .ASM" program published in the March and April 1986 issues, and in the February 1987 issue of "REMark" is the printing routines. Most printers, while having some download capability, just did not use the same kind of character set download capability as the Panasonic KX-P1092. Additionally, the routines were slow, used an extensive amount of disk space, and didn't have small letter reverse video capability. Finally, the draft font didn't have the dot density for well-formed graphics.

If your printer has a bit image graphics capability, the following information should assist you in getting full graphics capability out of "PAINT.ASM!"

The Fix!

I believe that more printers have a bit image graphics capability, where, like the downloaded characters, the printer is sent information on each character in the form of bytes that indicate which pins of the print head fire and which do not. The difference is that the information is not downloaded into the printer beforehand as a different character set. Our new routine looks up the appropriate information in a data matrix at the end of the program and sends the printing information directly. This eliminates the use of data files, printer memory limits and many of the special printer codes.

As discussed in the article "PAINT.ASM — Part 2" published in the April 1986 issue of "REMark", each character is composed of a pattern of '0's and '1's. '1's represent when a print head pin would fire and '0's represent blank areas. Unlike the downloaded character set used previously, there is no limit on how often or in what sequence the pins may fire. This increases the dot density, and hence, the character definition considerably.

Again, because the dot matrix printer uses a print head with a vertical row of pins, it prints each character from left to right, all appropriate pins firing at the same time, to form each vertical segment or column of the character. Each of these vertical columns requires one or two bytes of data.

Let's review the Panasonic's bit image graphics capability first, since this is the printer I have, then discuss a few other printers' capabilities.

The Panasonic KX-P1092 Multi-Mode Printer and others have 8- and 9-pin bit image mode selection. Of the 9 pins in the print head, the 8-pin bit image graphics mode uses the upper eight pins only, as shown in Figure 1. If only 7 pins were to be used for some reason, then the uppermost pin would not be used either. Each pin corresponds to a power of two. Therefore, by summing the powers of two corresponding to each of the pins you wish to fire, you will obtain a numerical value which instructs the printer to print one column of dots.

For example, suppose you wanted to fire pins 1, 2, 5, and 8, simultaneously. You would compute the sum of 128+64+8+1 for a total of 201 and send this value to the printer, which would then print a single column of dots corresponding to pins 1, 2, 5, and 8.

Pin Number	8-Bit Interface	7-Bit Interface	Power of 2
1	128	Not used	7th power
2	64	64	6th power
3	32	32	5th power
4	16	16	4th power
5	8	8	3rd power
6	4	4	2nd power
7	2	2	1st power
8	1	1	Ø power
9	Not used	Not used	7th power

Figure 1

In the 9-pin bit image mode, all 9 pins of the print head may be fired. The 9 pins in the print head are divided into two portions, the upper 8 pins, as shown in Figure 1, and the bottom pin. The firing of one or more of the upper 8 pins represents 1 byte of data, and the 9th pin represents a second byte of data -128 (the 7th power of 2). Together, these two bytes determine the dot configuration for a single column of graphics.

This time, if we wished to fire pins 1, 2, 5, 8, and 9, simultaneously, you would determine the sum of 128+64+8+1 for a total of 201, as before. This would be sent to the printer as byte #1. Byte #2, which is 128, would then also be sent, after which the printer would print a single column of dots corresponding to pins 1, 2, 5, 8, and 9 — a piece of cake, right?

Taking this information, we can form a complete character set very easily, much in the same manner as we did in my second article. Figure 2 provides some sample characters for the 9-pin bit image mode.

Study Figure 2 carefully. Notice that pin 9 is the leftmost pin, but becomes the second byte of information sent to the printer. The rightmost pin is pin 1 with a value of 128, with each pin going left from that one, descending in value.

"Ah", you say. "What does that have to do with the price of pork in Portugal?" Well, it is important if you wish to change your character set from mine. It is also important if you must modify Figure 2 to 7 or 8-pin operation, depending upon the capability of your printer. For example, for 8-pin operation, the reverse video graphics character `g' would be represented by the number sequence 255, 255, 237, 237, 297, 237, 237, 255, 255. Review Figure 2 again to see where those numbers came from. To maintain the character height to width ratio, you may wish to drop the last number (255), so each character is only 8 columns wide. This was easy — but recently I received a letter from an MPI PrintMate 99 owner. This printer uses only 6 pins to form graphics characters. Figure 3 shows its pin configuration for graphics:

```
REVERSE VIDEO CHARACTERS:
#                          $
111111111  = 255,128       111111111  = 255,128
111011011  = 219,128       111011011  = 219,128
100000000  = 0,128         110110001  = 141,128
100000000  = 0,128         100110101  = 172,128
111011011  = 219,128       100000000  = 0,128
100000000  = 0,128         110110100  = 45,128
100000000  = 0,128         110000101  = 161,128
111011011  = 219,128       111001011  = 211,128
111111111  = 255,128       111111111  = 255,128

GRAPHICS:
_                          b
000000001  = 128,0         000011000  = 24,0
000000011  = 192,0         000011000  = 24,0
000000111  = 224,0         000011000  = 24,0
000001111  = 240,0         111111111  = 255,128
000011111  = 248,0         111111111  = 255,128
000111111  = 252,0         000011000  = 24,0
001111111  = 254,0         000011000  = 24,0
011111111  = 255,0         000011000  = 24,0
111111111  = 255,128       000011000  = 24,0

REVERSE VIDEO GRAPHICS:
g                          h
111111111  = 255,128       111111111  = 255,128
111111111  = 255,128       111101111  = 247,128
110110111  = 237,128       111101111  = 247,128
110110111  = 237,128       111101111  = 247,128
110100011  = 197,128       100101001  = 148,128
110110111  = 237,128       110000011  = 193,128
110110111  = 237,128       111000111  = 227,128
111111111  = 255,128       111101111  = 247,128
111111111  = 255,128       111111111  = 255,128
```

Figure 2

Pin Number	6-Bit Interface	Power of 2
1	32	5th power
2	16	4th power
3	8	3rd power
4	4	2nd power
5	2	1st power
6	1	Ø power

Figure 3

For the MPI-99, graphics printing is accomplished using only the lower six pins of its 9-pin print head. Graphics print data is sent to the printer in eight-bit bytes where each byte represents one column of the character's dot pattern. The most significant bit is ignored by the printer. The next most significant bit must be a '1' to indicate graphics data. The lower 6 bits specify the dot configuration. If the next to the most significant bit is not a '1', but a '0', the byte is interpreted as a graphics termination character.

In order to fire pins 1, 3, 5, and 6, therefore, we would need to sum 32+8+2+1 and add 64, for a total of 107, to designate the character as graphics to the printer.

The character set of Figure 2 would need to be completely revised so each character is represented by a dot matrix 6x6. Since the normal character set is probably 7×9, this would make reverse video of the normal character set impossible with only one pass of the print head. A way around this problem may be to modify the program to permit printing the upper half of a line of characters, then printing the lower half on the next pass. With a mix of normal characters with graphics or reverse video, however, this may become near impossible.

Another possible solution would be to make the entire character set 6×6 — by adding the 95 normal characters to our data matrix and modifying our program to recognize these, instead of the printer's normal character set. The resulting graphics screen would take only about 2/3 of the normal page width and height, but normal text could then be printed with graphics and reverse video.

If one of you has found another solution to this problem, please let me know so I can pass it on.

Program Changes

Before going further in our discussion, let's make the program changes given in Listing 1. Make a copy of your current version of "PAINT.ASM" on your working disk — DO NOT USE YOUR ONLY COPY! Print out a copy of the program and carefully review your program's structure. NOTE ANY CHANGES YOU MADE from the original program THAT YOU WISH TO KEEP. After the enclosed revisions have been made to the program, you may have to modify your changes again.

The changes of Listing 1 are straight forward and follow the general organization of the original program. After the changes have been made, carefully review the program again looking for the obvious errors in structure — misplaced commas, semicolons, etc. — which usually cause most of the programming problems.

DON'T ASSEMBLE AND RUN YOUR NEW PROGRAM YET.

Other Printer Considerations

The program should work, as is, for printers that are similar to the Panasonic 1090 series. But, as you all know, printers are renowned for being NON-STANDARD, even from the same manufacturer! I only own the Heath H-14 and the Panasonic KX-P1092, so I know very little about other printers. I really appreciate those of you who have sent me information regarding your printers, but without the hands-on experience — it's like learning how to drive via correspondence courses. Anyway, through those friendly and considerate individuals who have sent me material, I've learned a few problem areas that may be of some assistance to others.

One problem area we've touched on — number of print head pins to fire — especially with the MPI. Another printer with similar problems is the Microline 92, which uses 7 pins to form characters. As we discussed, I recommended forming a whole new character set — using a single byte of information for each vertical column of dot information. In addition to deleting the two leftmost columns of each block of Figure 2, it appeared that the normal character width was 11 dots long, rather than the nine currently used. Two more lines had to be added to each character so the new characters matched the width of the normal characters. Take a close look at the characters formed by your printer, checking both height and width of the normal character set, so the character set you add will blend with your printer's present set.

Knowing the differences between your normal character set and the capabilities of your printer compared to mine, we need to make several changes to the revisions of Listing 1. You need to become intimate with your printer's operating manual for this — sleeping with it under your pillow isn't enough either.

The first thing we want to fix is the character codes at the end of the file. Generate a separate, complete file containing a character set composed of information, such as presented in Figure 2. This file is simply a worksheet to facilitate easy calculation or modification of each character's byte information and will not be used by "PAINT .ASM" in any way. Double check if you need 11 bytes of information of 1 byte per column of character dots or something else. You will need a full set of reverse video for your normal character set, then normal and reverse video graphics. When complete, you should have 95 reverse video printer characters, 33 normal graphics characters, and 33 reverse video graphics characters.

Starting at Reverse Character (RVCHAR) R1 of Listing 1, you can see that it is defined by the 18 bytes of information I needed. Replace this line with the number of bytes of new information you just made for each character. Do this for ALL the characters — RVCHAR, GCHAR (graphics char), and RVGCHR (reverse graphics char). REMEMBER TO SAVE YOUR WORK OCCASIONALLY — WITH BACKUP COPIES. The work you did on the character set and the changes to the program are too time consuming to have to do again.

Next, we need to check the printer codes. The codes of Listing 1 were for the Panasonic. The following lines were those that I recommended for the Microline 92 user and provide an example of those lines you may need to modify to match your printer's requirements. Again, check your printer manual for comparable codes (see the accompanying sidebar for an explanation of printer codes):

If your printer doesn't have EMPHASIS or DOUBLE strike printing (both are types of BOLDFACE printing), change these to reflect some other printer capability you may desire to use, i.e., elongated printing, script, etc. Modify MSG24 accordingly.

```
MSG33:
```

Needs to be adjusted to ensure the printer doesn't exit in a strange configuration — set default line spacing, release emphasized/enhanced printing, etc.

```
ADJ1 routine:
  Change MVI  B,18 to MVI  B,11
```

(Reflects the number of bytes of information per character.)

```
ADJ3 routine:
  Change MVI  B,18 to MVI  B,11
```

(Same reason.)

```
ADJ4 routine:
```

ADJ4 was to adjust the width of the new character set to the size of the normal character at a different pitch — if they differ. In Listing 1 the size of the character was modified by sending the last two bytes of information again. If your character set doesn't change with the pitch, or you are able to adjust the pitch (80/96 characters per line) of your characters by another means, the ADJ4 routine may be modified:

```
ADJ4    MOV     E,M       ;E=CHAR BYTE
        CALL    PRNTCH
        INX     H
        DCR     B
        JNZ     ADJ4      ;CHAR SENT
        LXI     H,MSG20   ;END GRAPHICS
        CALL    LPMSG
        POP     H
        INX     H
        JMP     PRLP
```

Delete the three lines of ADJ5. Remove the BITIMG routine entirely and replace with:

```
BITIMG  PUSH    D
        LXI     H,MSG19   ;BIT IMAGE SET
        CALL    LPMSG
        POP     D
        RET
```

Note that the last line on page 116 of the Microline 92 manual should read that Line

```
MSG19 DB 3,'$'     ;7-pin graphics mode
MSG20 DB 3,2,'$'   ;end graphics mode
MSG24
      DB '    3 = LPT Line Spacing of 7.',13,10
```

(This needs to be changed to reflect the different character heights.)

```
MSG25 DB 28,'$'    ;CPL=96 or 12 CPI
MSG26 DB 30,'$'    ;CPL=80 or 10 CPI
MSG27 DB 27,37,39,14,'$'  ;need line spacing of 7/72 (14/144?)
```

(You may need to play with this one.)

```
MSG28 DB 27,54,'$'  ;reset default line spacing
MSG29 thru 32:   Change these codes as you see fit.
```

60 switches the printer out of graphics mode.

The Microline 92 printer also had a feature that may exist on your printer. The Microline 92 manual on page 117 talks of Multiline Graphics. Since we are not printing pure graphics from our screens (we rely on the normal character set of the printer for all non-graphics, non-reverse video characters), we cannot use this function. You would have to redefine all your characters.

Another problem that you may experience, that I don't have an answer to, is in proportional printing — you may note a mismatch in character width between your printer's character and your defined graphics character. On the Microline 92, for example, the printer's normal character set formed a character five wide with one position turned off for character spacing. Other characters were six wide with one position off for character spacing. You might try adjusting your graphics character width, ADJ3, without changing your character set until you see the results. If it then works out well, modify your characters accordingly.

Remember that line spacing corrections will take care of the skipped lines between your lines of graphics, but it will also change the spacing between lines of normal text — which means descenders will overlap anything on the next line. The only fix is to minimize the mixture of graphics and normal text. For example, print letterhead separately, several pages at a time, then reinsert it into the printer to print the normal text. This works well, with only minor inconvenience.

Closing

Well, every good thing must come to an end, and so must this article. I hope that the changes and suggestions provided above assist you in better understanding your printer and its relationship to this program. Unlike the last time, where I said this completes my work on "PAINT.ASM", I can't say it this time — as my work is continuing.

At the chance of breaking some 8-bit hearts, I have procured an H-100 — much to my wife's chagrin, and over objections of "What are you going to do with three computers?" Computers are a little like rabbits, aren't they? Anyway, I have already modified the program to work on the H-100 and am now adding a color capability. I hope to write on these experiences, also. But, then, not to stop there, I am wrapped up in trying to get the program to link screens in such a manner as to permit the printing of a full page (8.5×11) of graphics and be able to display any part of that page on the screen for modification at any time — say, lines 11 through 34, for example. This final mod has me bogged down for the moment, but we'll see. If I get it to work, I'll write that up to.

Those of you who have been following along closely, or have written to me, may have noted that my address has changed. Like I said, time has been flying, and now it's my turn again. I've been transferred to Elizabeth City to resume my duties as a Coast Guard aviator, effective August 1st. As a result, I may be a little sloooow answering your mail, if I get it at all. Please leave a little extra time for a response, then write again if you don't hear from me — the first several months will be very hectic, between moving and retraining in an aircraft.

Again, I appreciate those people who have taken the time to write and send information on their printer. I remain very interested in any improvements readers feel may be appropriate, questions, or problems that are being caused by my program. Please include copies of pertinent printer info if you need help interfacing it with the program. Self-addressed, stamped envelopes are greatly appreciated.

I would be happy to send the current source code for $4.00 or the finished product on disk for $10.00, if you include a preformatted disk with your request. I generally use CDR or H/Z CP/M 2.2.4 double-sided, double-density, soft-sectored disks (H-100 compatible), but single-sided, single-density, hard-sectored disks are probably best. Please include a phone number, in case I have problems.

Understanding Your Printer's Escape Codes

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

I've received some requests that I explain printer escape codes — specifically, what is happening when I ask the readers to substitute their printer codes in place of mine in "PAINT.ASM." Let's take it from the top and discuss it.

Like the computer, most printers have considerably more capability than just displaying or printing words or characters — excluding the Heathkit H-14 printer (a plain vanilla, draft printer; 6 years old, but still doing fine). In addition to various dip switches or front panel switches to activate these capabilities, most printers can also be software controlled through a set of escape codes.

For those of you without a printer, or just thinking of purchasing a printer, let's take a look at the computer's operation manual. Chapter 11, in the old H-89 manual, or Appendix B of the H-100 Series User's Manual, has a section to discuss Heath Escape Sequences. These 'ESC'ape codes cover numerous functions, such as cursor movement, erasing and editing, power-up configuration, modes of operation, and others. Each of these codes are discussed briefly in the computer manual and I won't cover them again here. But, we do need to recognize that these codes are two or more bytes of information sent to the computer to control these functions.

For example, the Heath ESCape code to move the cursor home is "ESC-H". If E-S-C-H were typed as individual letters, however, the computer would not recog-

nize these as a code. There is a separate "ESC" key on the keyboard. By holding down the "ESC" key, while typing the "H", the cursor will return "Home." Upon receipt of an "ESC", the computer recognizes that it will have to act on, or do something special with, the next byte of information and looks for another keystroke.

Software control works the same way. Look at the beginning of the description of ASCII characters (ASCII — American Standard Code for Information Interchange) in the same section of the manual and you will find a separate line item for "ESC" or Escape. In computer language, the code is 033 in Octal, 27 in Decimal or 01B in Hex. The ASCII character "H" is 110 in Octal, 72 in Decimal or 048 in Hex. Upon receipt of the 2 byte code 27,72 in Decimal, the cursor will return "Home."

Codes are sent to the printer in the same manner, but the codes differ widely among printers. This is the reason good software packages generally have several different "printer drivers" — each meant to interpret escape codes differently for each type of printer. A typical printer has software commands to control line spacing, paper feed, page formats, data control, etc.

The Panasonic KX-P1092 dot matrix printer, for example, uses about 78 ESCape codes. A few of these are:

```
ESC-Ø           Sets line spacing to 1/8"
ESC-2           Sets line spacing to 1/6"
ESC-n           Sets NLQ print
ESC-N-n         Sets skip perforation to some number n
ESC-E           Sets emphasis print
ESC-F           Releases emphasis print
ESC-*-m-nl-n2   Selects 8-pin bit image mode
ESC-^-m-nl-n2   Selects 9-pin bit image mode
```

As you can see, these can get very complicated and are not limited to just a couple of bytes of information. But, these codes are necessary for broad, yet flexible, printer capability.

Each software language has a means of sending these codes to the printer. My word processor "MAGIC WAND" uses the "\OUT" command (e.g., \OUT27,45,1\ sets underlining on my printer). BASIC uses the "OUT" command in a similar manner. In my program "PAINT.ASM", we send the information as we would a message; e.g., MSG29 DB 27,69,'$' (which sets "emphasis" mode). A person with a different printer must check the capabilities of his printer, select the appropriate code, and replace my printer code with his own. MSG29 for his printer may need to read MSG29 DB 27,53,'$' which might set "Boldface" on that printer. The codes may change, but generally, the method of sending the codes to the printer should remain the same. ✱

Listing 1
Make the following changes to the "PAINT.ASM" program presented in the "PAINT.ASM" articles in the March '86, April '86, and February '87 issues of "REMark":

```
IN BLUE ROUTINE:  Change MSG51 to MSG36.
        LXI     D,MSG36 ;MOVE TO MARK 2
IN MARK2 ROUTINE:  Change MSG52 to MSG37.
        LXI     D,MSG37

COLCT1  DCR     C
        JZ      COLCT3
        MOV     A,M     ;GET CHAR
        CPI     ESCAPE
        CZ      COLCT2
        INX     H
        DCR     B
        JMP     COLCT1
```

IN NEGERR ROUTINE: Change MSG53 to MSG38.

```
NEGERR  LXI     D,MSG38 ;NEG RESULT
```

DELETE ALL of the following old routines: CHFONT, QFONT, DRAFTD, AGAIN, NOFILE, LDFCB, LDFONT, DFTSET, GETFONT, RDFONT, RDLOOP, NEXTCH, FONTDN, RDRECD, BIT0, BIT1, BITSET, LPWORD, SETROM, LDROM, DRAFTR, PROPR, NLQR, LPLS, LS7, LS8, LS9, LS12, SETLS, PRNTR, PRLP, PRNTIT, and INCH.

REPLACE with the following routines:

```
CHFONT  LDA     MODLPT
        ORA     A       ;LPT SERIAL?
        JZ      INPUT   ;YES - SKIP
        LXI     D,MSG3  ;DUMP SCREEN TO MEMORY
        CALL    MSG50
        CALL    SAVE
        LXI     D,MSG9  ;DISABLE LINE 25
        CALL    PMSG

QFONT   LXI     D,MSG24 ;LPT CHARACTERISTICS
        CALL    PMSG
QUEST   LXI     D,MSG34
        CALL    PMSG
        MVI     C,CONIN
        CALL    BDOS    ;A=CHAR
        CPI     Ø31H
        JZ      ELITEP  ;SET ELITE
        CPI     Ø32H
        JZ      PICAP   ;SET PICA
        CPI     Ø33H
        JZ      LS9     ;SET LPT LS
        CPI     Ø34H
        JZ      LS12    ;SET LPT LS (DEFAULT)
        CPI     Ø35H
        JZ      EMPHS   ;SET EMPHASIS
        CPI     Ø36H
  JZ      EMPHR   ;RELEASE EMPHASIS
        CPI     Ø37H
        JZ      DBLS    ;SET DOUBLE PRINT
        CPI     Ø38H
        JZ      DBLR    ;RELEASE DOUBLE PRINT
        CPI     Ø39H
        JZ      DSPLY   ;NO CHANGE/DONE
        JMP     QUEST
ELITEP  MVI     A,1
        STA     PICAFG  ;SET ELITE PITCH
        LXI     H,MSG25
        JMP     SETLS

PICAP   MVI     A,Ø
        STA     PICAFG  ;SET PICA PITCH
        LXI     H,MSG26
        JMP     SETLS

LS9     LXI     H,MSG27
        JMP     SETLS
```

```
LS12    LXI     H,MSG28
SETLS   CALL    LPMSG
        LXI     D,MSG35 ;SET MSG
        CALL    PMSG
        JMP     QUEST

EMPHS   LXI     H,MSG29
        JMP     SETLS

EMPHR   LXI     H,MSG30
        JMP     SETLS

DBLS    LXI     H,MSG31
        JMP     SETLS

DBLR    LXI     H,MSG32
        JMP     SETLS

PRNTR   LXI     D,MSG5  ;PRINT SCREEN MSG
        CALL    MSG50
        CALL    SAVE
        LXI     H,BUFR
PRLP    MOV     A,M     ;GET CHAR
        CPI     15Q     ;CR? MEANS END OF SCREEN
        JZ      LPDONE
        MOV     E,A
        CPI     ESCAPE
        JZ      ADD1    ;ALLOW FOR ESC CODES
        LDA     COUNT
        DCR     A
        STA     COUNT
        CZ      PCRLF   ;RESET COUNT IF 0
        LDA     LPRVFG
        ORA     A
        JNZ     RVIDC
        LDA     LPGFFG
        ORA     A
        JNZ     GRAFC
        CALL    PRNTCH
        INX     H
        JMP     PRLP

PRNTCH  PUSH    PSW
        PUSH    H
        PUSH    B
        MVI     C,LSTOUT ;E=CHAR
        CALL    BDOS
        POP     B
        POP     H
        POP     PSW
        RET
```

Modify the following routines to:

```
ADD1    INX     H
        MOV     A,M
        CPI     070H    ;"p"?
        JZ      LPRVID
        CPI     071H    ;"q"?
        JZ      ELPRVID
        CPI     046H    ;"F"?
        JZ      LPGRAF
        CPI     047H    ;"G"?
        JZ      ELPGRAF
        INX     H
        JMP     PRLP

LPRVID  MVI     A,1
        STA     LPRVFG  ;RV FLAG
        INX     H
        JMP     PRLP

ELPRVID MVI     A,0
        STA     LPRVFG
        INX     H
        JMP     PRLP

LPGRAF  MVI     A,1
        STA     LPGFFG  ;GRAPHICS FLAG
        INX     H
        JMP     PRLP

ELPGRAF MVI     A,0
        STA     LPGFFG
        INX     H
        JMP     PRLP
```

DELETE the ADD96 and ADD129 routines, and ADD the following lines in their place:

```
RVIDC   PUSH    H
        CALL    BITIMG  ;BIT IMAGE GRAPHICS SET
        LDA     LPGFFG
        ORA     A
        JNZ     RVGC    ;RVID GRAPHICS CHAR
        LXI     H,RVCHAR
        MOV     A,E
        SUI     01FH    ;SUBTRACT 31
        MOV     C,A     ;C=NUMBER OF CHAR
        JMP     ADJ1

RVGC    LXI     H,RVGCHR
        JMP     GRAF1
GRAFC   PUSH    H
        CALL    BITIMG
        LXI     H,GCHAR
GRAF1   MOV     A,E
        SUI     05DH    ;SUBTRACT 93
        MOV     C,A     ;C=NUMBER OF CHAR

ADJ1    DCR     C
        JZ      ADJ3    ;COUNT ADJUSTED
        MVI     B,18
ADJ2    INX     H
        DCR     B
        JNZ     ADJ2
        JMP     ADJ1

ADJ3    MVI     B,18
ADJ4    MOV     E,M     ;E=CHAR BYTE
        CALL    PRNTCH
        INX     H
        DCR     B
        JNZ     ADJ4
        LDA     PICAFG
        ORA     A
        JZ      ADJ5    ;0=PICA
        DCX     H       ;ELITE PITCH, SEND LAST
        DCX     H       ;  2 BYTES AGAIN
        MOV     E,M
        CALL    PRNTCH
        INX     H
        MOV     E,M
        CALL    PRNTCH
ADJ5    POP     H
        INX     H
        JMP     PRLP

BITIMG  PUSH    D
        LDA     PICAFG
        ORA     A
        JNZ     IMG1
        LXI     H,MSG19 ;BIT IMAGE SET - PICA
 JMP     IMG2
IMG1    LXI     H,MSG20 ;BIT IMAGE SET - ELITE
IMG2    CALL    LPMSG
        POP     D
        RET
```

One line before the CKL Routine (last line of FILNM Routine), make the value of the B counter 14 vice 12. This will correct a problem with the File Name length. Also, after doing a last check for the decimal in the

FILIT Routine (CPI 02EH), replace the line JNZ LNERR ;LENGTH TOO LONG with the following lines:

```
        JZ      FILTYP
        CPI     ' '     ;BLANK?
        RZ              ;END OF NAME
        JMP     LNERR   ;TOO LONG
```

MODIFY the following routines to:

```
NEXTLN  LHLD    BPTRL
        SHLD    BPTRM   ;SET NEXT CHAR TO START OF LINE
        MVI     A,15Q   ;CR?
        CMP     M
        JZ      ENDLP   ;IF CR WE ARE DONE - CLEANUP
        MVI     A,80    ;# CHARS IN A LINE
        MOV     C,A
        STA     COUNT
FNDLN   LHLD    BPTRL
        MVI     A,ESCAPE
        CMP     M
        JNZ     FNDLN4  ;NOT AN ESC
        INR     C
        INR     C       ;ADD 2 TO ALLOW FOR ESC
        INX     H
        MOV     A,M     ;GET NEXT CHAR
        CPI     070H    ;'p'
        JZ      FNDLN2
        CPI     071H    ;'q'
        JZ      FNDLN3
FNDLN1  INX     H
        SHLD    BPTRL
        JMP     FNDLN

FNDLN2  MVI     A,1
        STA     RVFLAG  ;ESC p FOUND - SET RVFLAG
        JMP     FNDLN1

FNDLN3  XRA     A
        STA     RVFLAG  ;ESC q FOUND - 0 RVFLAG
        JMP     FNDLN1

FNDLN4  INX     H
        SHLD    BPTRL
        LDA     COUNT
        DCR     A
     STA     COUNT
        JZ      BACKUP  ;END OF LINE - BACKOFF BLANKS
        JMP     FNDLN

BACKUP  LDA     RVFLAG
        ORA     A
        JNZ     DISKIT  ;RV IN PROGRESS - BLANKS COUNT
        DCX     H
        MVI     A,' '
        CMP     M
        JNZ     DISKIT  ;LAST NON-BLANK FOUND
        DCR     C
        JZ      DCRLF   ;NO NON-BLANKS - PUT OUT CRLF
        JMP     BACKUP
```

```
DISKIT  LHLD    BPTRM
        MOV     A,M
        INX     H
        SHLD    BPTRM
        CALL    DISKOUT
        DCR     C
        JNZ     DISKIT
DCRLF   MVI     A,00DH  ;CR
        CALL    DISKOUT
        MVI     A,00AH  ;LF
        CALL    DISKOUT
        JMP     NEXTLN
```

ADD the following routine before the SCRLF Routine:

```
PCRLF   PUSH    H
        PUSH    D
        MVI     E,00DH  ;CR
        CALL    PRNTCH
        MVI     E,00AH  ;LF
        CALL    PRNTCH
        POP     D
        POP     H
        JMP     SETCT
```

MODIFY TODISK Routine as follows:

```
TODISK  LXI     H,DMA
        SHLD    DMAPTR  ;RESET TO BEGIN
        PUSH    B
        MVI     C,WRTFIL
        LXI     D,DFCB
        CALL    BDOS
        POP     B
        ORA     A       ;0?
        RZ              ;GOOD WRITE
        LXI     D,MSG12
        CALL    MSG50   ;NO MORE ROOM
        JMP     DKDONE
```

MODIFY the following messages as appropriate; some are repeated for clarity:

```
MSG19   DB  27,94,6,9,0,'$' ;9-PIN BIT IMAGE MODE (6)
MSG20   DB  27,94,1,10,0,'$' ;9-PIN BIT IMAGE MODE (1)
MSG21   DB  27,69,'$' ;ESC-E
MSG22   DB  27,72,'Normal:   $'
MSG23   DB  'Graphics: ',27,70,'$'
MSG24   DB  27,69,'SELECTION OF PRINTER CHARACTERISTICS:',13,10
        DB  '    1 = ELITE Pitch (96 cpl).',13,10
        DB  '    2 = PICA Pitch (80 cpl).',13,10
        DB  '    3 = LPT Line Spacing of 9.',13,10
        DB  '    4 = LPT Default LS of 12 (value of 1/6 inch).',13,10
        DB  '    5 = Set EMPHASIS MODE',13,10
        DB  '    6 = Release EMPHASIS MODE',13,10
        DB  '    7 = Set DOUBLE PRINTING MODE',13,10
        DB  '    8 = Release DOUBLE PRINT MODE',13,10
        DB  '    9 = No Change/Done.',13,10,'$'
MSG25   DB  27,77,'$' ;CPL=96
MSG26   DB  27,80,'$' ;CPL=80
```

```
MSG27   DB  27,48,'$' ;LS=9
MSG28   DB  27,65,12,27,50,'$' ;LS=12
MSG29   DB  27,69,'$' ;SET EMPHASIS
MSG30   DB  27,70,'$' ;REL EMPHASIS
MSG31   DB  27,71,'$' ;SET DOUBLE
MSG32   DB  27,72,'$' ;REL DOUBLE
MSG33   DB  27,65,12,27,70,27,72,'$' ;DEFAULT LPT EXIT
MSG34   DB  'What is the number of your selection <1 - 9>? $'
MSG35   DB  ' SET.',13,10,'$'
MSG36   DB  'MOVE CURSOR TO SECOND MARK.  CTRL-B WILL RESET.',27,121,53,'$'
MSG37   DB  'MOVE CURSOR TO NEW LOCATION.  CTRL-B WILL RESET.',27,121,53,'$'
MSG38   DB  13,10,'BLOCK INCORRECTLY DEFINED.$'
```

Note: Your printer codes may be different from the above. See text.

DELETE all MSGs above MSG38 and the following Codes: INT, CHK, HZDRAF, FTWORD, DRAFTF, and FONTFG

ADD ALL of the following Code lines before the BUFR and STACK space: (Be VERY CAREFUL in entering the following information, as it determines what will be printed when a graphics or reverse video character is sent to the printer. Ensure the proper number of data elements are typed — 18 for each character and a 9-pin bit image capable printer.

```
PICAFG  DB      0       ;0=PICA, 1=ELITE PITCH
;
; Following codes are for reverse video printer characters:
RVCHAR
R1   DB 255,128,255,128,255,128,255,128,255,128,255,128,255,128,255,128,255,128
R2   DB  255,128,255,128,255,128,2,128,2,128,255,128,255,128,255,128,255,128
R3   DB  255,128,255,128,15,128,15,128,255,128,15,128,15,128,255,128,255,128
R4   DB  255,128,219,128,0,128,0,128,219,128,0,128,0,128,219,128,255,128
R5   DB  255,128,219,128,141,128,172,128,0,128,45,128,161,128,211,128,255,128
R6   DB  255,128,156,128,153,128,243,128,231,128,207,128,153,128,57,128,255,128
R7   DB  255,128,241,128,0,128,14,128,110,128,4,128,241,128,228,128,255,128
R8   DB  255,128,255,128,255,128,47,128,15,128,31,128,255,128,255,128,255,128
R9   DB  255,128,255,128,195,128,129,128,24,128,60,128,126,128,255,128,255,128
R10 DB  255,128,255,128,126,128,60,128,24,128,129,128,195,128,255,128,255,128
R11 DB 255,128,255,128,219,128,231,128,129,128,231,128,219,128,255,128,255,128
R12 DB 255,128,247,128,247,128,193,128,193,128,247,128,247,128,255,128,255,128
R13 DB  255,128,255,128,255,128,255,0,248,0,248,128,255,128,255,128,255,128
R14 DB 255,128,247,128,247,128,247,128,247,128,247,128,247,128,255,128,255,128
R15 DB 255,128,255,128,255,128,252,128,252,128,252,128,255,128,255,128,255,128
R16 DB  255,128,252,128,249,128,243,128,231,128,207,128,159,128,63,128,255,128
R17 DB  255,128,129,128,56,128,114,128,102,128,78,128,28,128,129,128,255,128
R18 DB  255,128,255,128,190,128,0,128,0,128,254,128,255,128,255,128,255,128
R19 DB  255,128,190,128,124,128,120,128,114,128,102,128,14,128,158,128,255,128
R20 DB  255,128,189,128,126,128,110,128,110,128,110,128,0,128,145,128,255,128
R21 DB  255,128,243,128,227,128,203,128,155,128,0,128,0,128,251,128,255,128
R22 DB  255,128,29,128,30,128,94,128,94,128,92,128,65,128,99,128,255,128
R23 DB  255,128,129,128,0,128,110,128,110,128,110,128,32,128,177,128,255,128
R24 DB  255,128,124,128,121,128,115,128,103,128,79,128,31,128,63,128,255,128
R25 DB  255,128,145,128,0,128,110,128,110,128,110,128,0,128,145,128,255,128
R26 DB  255,128,157,128,12,128,110,128,110,128,110,128,0,128,129,128,255,128
R27 DB 255,128,255,128,255,128,147,128,147,128,147,128,255,128,255,128,255,128
R28 DB 255,128,255,128,255,128,146,128,144,128,145,128,255,128,255,128,255,128
R29 DB  255,128,255,128,231,128,195,128,153,128,60,128,126,128,255,128,255,128
R30 DB 255,128,215,128,215,128,215,128,215,128,215,128,215,128,255,128,255,128
R31 DB  255,128,255,128,126,128,60,128,153,128,195,128,231,128,255,128,255,128
R32 DB  255,128,159,128,63,128,127,128,114,128,114,128,7,128,143,128,255,128
R33 DB  255,128,129,128,60,128,66,128,66,128,90,128,2,128,130,128,255,128
R34 DB  255,128,224,128,192,128,155,128,59,128,155,128,192,128,224,128,255,128
R35 DB  255,128,0,128,0,128,110,128,110,128,44,128,0,128,145,128,255,128
R36 DB  255,128,195,128,129,128,60,128,126,128,126,128,60,128,60,128,255,128
R37 DB  255,128,0,128,0,128,126,128,126,128,60,128,129,128,195,128,255,128
R38 DB  255,128,0,128,0,128,110,128,110,128,110,128,44,128,60,128,255,128
R39 DB  255,128,0,128,0,128,110,128,111,128,111,128,63,128,63,128,255,128
R40 DB  255,128,129,128,0,128,126,128,126,128,118,128,48,128,48,128,255,128
R41 DB  255,128,0,128,0,128,110,128,239,128,110,128,0,128,0,128,255,128
R42 DB  255,128,255,128,126,128,0,128,0,128,126,128,255,128,255,128,255,128
R43 DB  255,128,253,128,252,128,126,128,0,128,1,128,127,128,255,128,255,128
R44 DB  255,128,0,128,0,128,231,128,195,128,153,128,60,128,126,128,255,128
R45 DB  255,128,0,128,0,128,126,128,254,128,254,128,252,128,252,128,255,128
R46 DB  255,128,0,128,0,128,143,128,231,128,143,128,0,128,0,128,255,128
R47 DB  255,128,0,128,0,128,143,128,231,128,241,128,0,128,0,128,255,128
R48 DB  255,128,129,128,0,128,60,128,126,128,60,128,0,128,129,128,255,128
R49 DB  255,128,0,128,0,128,111,128,111,128,111,128,15,128,159,128,255,128
R50 DB  255,128,129,128,0,128,60,128,126,128,56,128,0,128,130,128,255,128
R51 DB  255,128,0,128,0,128,103,128,99,128,105,128,12,128,158,128,255,128
R52 DB  255,128,157,128,12,128,110,128,110,128,110,128,32,128,177,128,255,128
R53 DB  255,128,63,128,127,128,0,128,0,128,127,128,63,128,255,128,255,128
R54 DB  255,128,1,128,0,128,254,128,254,128,254,128,0,128,1,128,255,128
R55 DB  255,128,31,128,7,128,240,128,252,128,240,128,7,128,31,128,255,128
R56 DB  255,128,1,128,0,128,249,128,227,128,249,128,0,128,1,128,255,128
R57 DB  255,128,60,128,153,128,195,128,231,128,195,128,153,128,60,128,255,128
R58 DB  255,128,31,128,199,128,240,128,240,128,199,128,31,128,255,128,255,128
R59 DB  255,128,60,128,56,128,114,128,102,128,78,128,28,128,60,128,255,128
R60 DB  255,128,255,128,0,128,0,128,126,128,126,128,126,128,255,128,255,128
R61 DB  255,128,63,128,159,128,207,128,231,128,243,128,249,128,252,128,255,128
R62 DB  255,128,255,128,126,128,126,128,126,128,0,128,0,128,255,128,255,128
R63 DB  255,128,255,128,207,128,159,128,63,128,159,128,207,128,255,128,255,128
R64 DB 254,128,254,128,254,128,254,128,254,128,254,128,254,128,254,128,254,128
R65 DB  255,128,255,128,255,128,31,128,15,128,47,128,255,128,255,128,255,128
R66 DB 255,128,253,128,232,128,234,128,234,128,234,128,225,128,240,128,255,128
R67 DB  255,128,0,128,0,128,238,128,238,128,238,128,224,128,241,128,255,128
R68 DB 255,128,241,128,224,128,238,128,238,128,238,128,238,128,246,128,255,128
R69 DB  255,128,241,128,224,128,238,128,238,128,238,128,0,128,0,128,255,128
R70 DB 255,128,241,128,224,128,234,128,234,128,234,128,226,128,247,128,255,128
R71 DB  255,128,239,128,128,128,0,128,111,128,127,128,63,128,191,128,255,128
R72 DB  255,128,243,128,225,0,237,0,237,0,237,0,224,0,224,128,255,128
R73 DB  255,128,0,128,0,128,239,128,239,128,239,128,224,128,240,128,255,128
R74 DB  255,128,255,128,238,128,32,128,32,128,254,128,255,128,255,128,255,128
R75 DB  255,128,255,128,254,128,255,0,255,0,32,0,32,128,255,128,255,128
R76 DB  255,128,0,128,0,128,243,128,227,128,201,128,156,128,254,128,255,128
R77 DB  255,128,255,128,126,128,0,128,0,128,254,128,255,128,255,128,255,128
R78 DB 255,128,224,128,224,128,239,128,240,128,239,128,224,128,240,128,255,128
R79 DB 255,128,224,128,240,128,231,128,239,128,231,128,224,128,240,128,255,128
R80 DB 255,128,241,128,224,128,238,128,238,128,238,128,224,128,241,128,255,128
R81 DB  255,128,224,0,224,0,237,128,237,128,237,128,225,128,243,128,255,128
R82 DB  255,128,243,128,225,128,237,128,237,128,237,128,224,0,224,0,255,128
R83 DB 255,128,224,128,224,128,247,128,239,128,239,128,247,128,255,128,255,128
```

Continued on Page 84

Leisure Suit Larry

Jennifer Louches
Heath Company
St. Joseph, MI 49085

So, you can't afford that trip to Las Vegas this winter? Has Sierra got a deal for you! For just $39.95 and "proof" that you're 21 or older, you can travel with Leisure Suit Larry to the sin city of Lost Wages.

For all of you fans of King's Quest and Space Quest, Sierra has come up with a brand new 3-D animated adventure. "Leisure Suit Larry in the Land of the Lounge Lizards" is not for adventure players who are easily offended. In writers Al Lowe and Mark Crowes' own words, "If it's silly, rude, dirty or funny, we probably thought of it while writing this program." They go on to add that "some of the puzzles in this game may require deranged thinking." In playing Leisure Suit Larry, I found this to be an accurate assessment of the game.

For those of you who have kids, you'll be happy to know that the authors of this game have inserted a five question quiz at the beginning of the game designed to boot anyone who's under 21 out of the program. The questions are difficult (I'm 27 and I was told that I "couldn't possibly be over 21" on more than one occasion) and very entertaining. They're selected randomly from a pool of questions, so even if you make it in one time, it's no guarantee that you'll get the same questions the next time around. Sure, most kids will manage to figure out the answers over a period of time, but on their first few tries, I doubt that they'll be allowed to play.

Once you're past the quiz, you find yourself as Leisure Suit Larry standing outside Lefty's Bar. If you stand out on the sidewalk long enough, a nice little dog comes by and mistakes you for a fire hydrant. This is the perfect introduction to the type of situations you'll be running into later on in the game.

You open the door and walk into Lefty's in your "usual studly manner." Well, you get the idea. In the city of Lost Wages there are bars, a casino (of course), a convenience store, ladies of the evening, flashers and much, much more. It's just like a real trip to Las Vegas only you won't lose the shirt off you're back or get a social disease.

Equipment Needed To Play

You can play Leisure Suit Larry on any PC or PC compatible computer. I played the game on a Z-248 with an EGA card and on a ZFL-181. I definitely recommend playing on a machine equipped with an EGA card. The game is much more interesting and easier to play when you can make out details in the environment. On the '181, you simply couldn't see or determine what a lot of the objects were. As with any adventure game, to win you have to pay close attention to details. That moosehead on the wall in the bar just might be an important clue.

Basic Interaction

To move Larry through the town of Lost Wages, you can either use a joystick or the pad/arrow keys on your keyboard. Larry can move north, south, east, west and diagonally. During the game, you can change from joystick to keyboard or vice versa by simply stopping Larry's movement and starting again on the new device.

You can talk to characters you come across during the game or give commands by using a single key word or simple sentences followed by a return. Some key words that I found helpful are listed below.

blow up	eat	order
buy	get	read
change	give	take
count	jump	undress
cut	kiss	use
drink	look	wear
drop	open	

Probably the most important word you'll use is "look." By looking at your environment frequently, you'll find hints that you would otherwise miss.

Another helpful word is "use." You can use most any object you run across if you're in the right place at the right time. And remember that most objects won't be provided if they aren't meant to be used in some manner. In Leisure Suit Larry it pays to be a packrat. Save just about anything you come across, no matter how bizzare. Sometime later in the game you may wish you had a television remote control.

If you're having problems with Larry moving too fast (who could believe Larry would

move too fast), you can slow down the animation. Or, if you really want to get Larry moving, you can even speed up the animation. To change the speed type "slow" or "fast" and press ENTER. To return to your original speed, type "normal" and press ENTER.

Function and control keys can also be used in Leisure Suit Larry. They help to cut time and save your fingers from repetitious typing. Function and control keys are assigned as follows:

F1	Help: shows list of control keys
F2	Toggle sound on/off
F3	Echo (repeat) previous command
F5	Save game
F7	Restore game
F9	Restart game
F10	Changes speed
CTRL-6	Stops game introduction
CTRL-C	Cancel typed input
CTRL-J	Reset joystick
CTRL-R	Toggle RGB/composite graphics mode
ALT-Z	Quit game
TAB	Inventory

CTRL-ALT (right or left arrow) Shift display

Leisure Suit Larry also contains a series of pop-up menus. To access the pop-up menus, press ESC. You can get into the pop-up menus at any time, even if your keyboard is "frozen."

The first pop-up menu appears as a small square in the upper left-hand corner of your screen. A listing of menus is provided along the top edge of the screen. To access subsequent menus, use the left and right arrow keys. Once in the menu you want, use the up and down arrow keys to move to your selection. Press ENTER to select a menu item. To exit the pop-up menus, press ESC.

The following is a listing of pop-up menus and items included in each.

Sierra

Help	This menu gives you a short summary of commands and functions.
About Larry	Gives a status on how long you've been playing.
Calculator	Strictly amusement.
Puzzle	Same as Calculator.

File	**Action**	
Save <F5>	Inventory	Inventory of what you're carrying
Restore <F7>		
Restart <F9>	See Object	Allows you to examine any object in your inventory.
Quit <ALT-Z>		
	Bodily Function	Lets you perform any bodily function you specify.

Special

Sound off <F2>	
Graphics mode <CTRL-R>	
Joystick <CTRL-J>	
Clock on/off	Toggles on-screen clock.
Boss key <CTRL-B>	
Pause Game	Allows you to pause action.

Speed

Normal
Slow
Fast
Fastest
Change <F10>

Saving And Restoring Games

In order to let your creativity go unhindered and to save you the frustration of starting a game over, I would suggest that you save your game frequently. You can never tell what may be waiting for you around the next corner. Being too nervous about getting wiped off the face of the earth never helped any good adventurer.

So, to save a game, type "save game" and press ENTER or press F5 to save your current game location. You can save a game at any time during play.

Once you type "save game" or F5, you'll receive a prompt asking which directory to save your game location. Just press ENTER and follow the prompts. All you'll have to do is substitute a blank formatted disk for your game disk and press ENTER.

If you're using a machine with two disk drives, backspace through the suggested \ and enter b: instead. This will cause your game to be saved to the formatted blank disk that you have so wisely remembered to insert in drive b at the beginning of the game.

When using a hard disk, you need only press ENTER when prompted about the directory in which to save your game position.

To restore a game, type "restore game" and press ENTER, or you can simply press F7. You'll be asked where you want to save the game. If you want to accept the default directory, press ENTER. Otherwise, type CTRL-C and type the drive or directory on which you saved the game and press ENTER. Now you can select the game you wish to restore by moving the pointer and pressing ENTER.

Some Specifics

Now that you know the general ways to get around, talk and save or restore games, there are some specific commands that will be useful when playing Leisure Suit Larry.

When you're in the Lost Wages casino, you have your choice of playing slots or blackjack. The slots and blackjack games are basically subprograms within the adventure game. They work pretty much like any other computer blackjack or slot machine program.

To play slots, move Larry over to the open slot machine and type "play slots". Your screen changes from the casino to a full screen slot machine. Directions and an on-screen menu are provided to help you with wagers, rules and what constitutes a winning combination. When you're finished losing money to the "one-armed bandit", type "exit slots".

Next, you can move to the blackjack table. The object of blackjack is to get as close to 21 as possible without going over. Face cards count as 10 points each, and aces count as 11 or 1, whichever is to your advantage. You are dealt two cards to begin with.

Now for a crash course in blackjack. When you first sit down at the blackjack table type "bet". You'll be asked to specify an amount that must be in $2.00 increments. After you specify your wager, you'll be dealt two cards. If you don't like the cards, you can "surrender" and half of your wager will be returned. If you think you're close enough to 21 to win with the two cards you're dealt, you can "stand". If you're dealt low cards, and need more to add up to 21, you can take a "hit". If the dealer shows an "ace up", you're asked if you want insurance. Insurance equals half of your wager and should be used with care. If the dealer has blackjack (21), you will be paid off at 2 to 1 odds. You are allowed to "double down" if your cards show a value of 10 or 11. If you opt to "double down", you are dealt one card and your bet is doubled. To quit playing type "stand up" at the end of a hand.

Catch all of that? If not, there's a help menu contained in the blackjack program and information on the reference card included with the game.

I found the best way to survive the casino was to save the game frequently while I was winning. Otherwise, Larry ended up penniless and eating out of trash cans for the rest of his wretched life. You will need the money you win in the casino to get through other portions of the game so brush up on your blackjack and hope that you're lucky at slots.

Exploring

There are a lot of interesting places to see and people to do in the city of Lost Wages. Make sure you explore an area thoroughly. Be friendly and talk to the people you meet. Ask them questions about themselves and the area you find them in. Some of the most unlikely people may be the most helpful. Sometimes it pays to be persistent, especially with women.

I had a lot of fun playing Leisure Suit Larry. It provided more than a few laughs and was quite challenging as adventure games go. When playing the game, I had the best time when I was playing it along with someone else. You get to share the laughs and another person may think of angles to the game that you may not. But, no matter if you play Leisure Suit Larry by yourself or with a friend, you'll have to have a sense of humor that's a little on the unusual side.

Remember, this is an adventure game and Larry can say or do any outrageous thing he wants. So let your imagination run and have fun playing Leisure Suit Larry in the Land of the Lounge Lizards.

(P.S. Don't forget to flush the toilet!)

Leisure Suit Larry in the Land of the Lounge Lizards $39.95
Sierra On-Line, Inc.
Coarse Gold, CA 93614
Customer Support (209) 683-6858
Bulletin Board (209) 683-4463 (They promise to be discreet.)
To get Sierra's hint book, call or write for information.

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

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

HEPCAT is here! *HEPCAT* is here! *HEPCAT* is here! So what is *HEPCAT*, you may ask? Why it's just another Pat Swayne SUPER-UTILITY. *HEPCAT* is an acronym for *HUG Engineer's and Programmer's CAlculation Tool*. Just what we don't need, another memory resident calculator, right? Wrong! With *HEPCAT*, you can throw away the rest and use the best. *HEPCAT* only uses two partial lines on your screen, and best of all, does NOT cause existing programs to stop executing! That means, while your computer is grinding numbers internally, you can be grinding them externally. Order ***HUG P/N 885-3045-37***.

Can't remember how to use the MS-DOS 'COPY' command? Forget the exact command line format for 'ASGNPART'. Too far to go for the MS-DOS manuals on the shelf on the other side of the room? Why not just type 'HELP' on the keyboard? You say it comes back with "Bad command or file name"? It wouldn't if you had HUG's ***HELP*** program. With *HELP* installed on your hard disk, all you need to do is type 'HELP' for a complete list of MS-DOS commands and transients along with a brief explanation of how each command works, as well as the format for its use. *HELP*, ***HUG P/N 885-8040-37***, works on *ALL* Heath/Zenith computers that run MS-DOS!

HUG Price List

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
H8 — H/Z-89/90				
ACCOUNTING SYSTEM	885-8047-37	CPM	BUSINESS	20.00
ACTION GAMES	885-1220-[37]	CPM	GAME	20.00
ADVENTURE	885-1010	HDOS	GAME	10.00
ASCRITY	885-1238-[37]	CPM	AMATEUR RADIO	20.00
AUTOFILE (Z80 ONLY)	885-1110	HDOS	DBMS	30.00
BHBASIC SUPPORT PACKAGE	885-1119-[37]	HDOS	UTILITY	20.00
CASTLE	885-8032-[37]	HDOS	ENTERTAINMENT	20.00
CHEAPCALC	885-1131-[37]	HDOS	SPREADSHEET	20.00
CHECKOFF	885-8010	HDOS	CHECKBOOK SOFTWARE	25.00
DEVICE DRIVERS	885-1105	HDOS	UTILITY	20.00
DISK UTILITIES	885-1213-[37]	CPM	UTILITY	20.00
DUNGEONS & DRAGONS	885-1093-[37]	HDOS	GAME	20.00
FLOATING POINT PACKAGE	885-1063	HDOS	UTILITY	18.00
GALACTIC WARRIORS	885-8009-[37]	HDOS	GAME	20.00
GALACTIC WARRIORS	885-8009-[37]	CPM	GAME	20.00
GAMES 1	885-1029-[37]	HDOS	GAMES	18.00
HARD SECTOR SUPPORT PACKAGE	885-1121	HDOS	UTILITY	20.00
HDOS PROGRAMMERS HELPER	885-8017	HDOS	UTILITY	16.00
HOME FINANCE	885-1070	HDOS	BUSINESS	18.00
HUG DISK DUPLICATION UTILITIES	885-1217-[37]	CPM	UTILITY	20.00
HUG SOFTWARE CATALOG	885-4500	VARIOUS	PRODUCTS THRU 1982	9.75
HUGMAN & MOVIE ANIMATION	885-1124	HDOS	ENTERTAINMENT	20.00
INFO. SYSTEM AND TEL. & MAIL SYSTEM	885-1108-[37]	HDOS	DBMS	30.00
LOGBOOK	885-1107-[37]	HDOS	AMATEUR RADIO	30.00
MAPLE	885-8005	HDOS	COMMUNICATION	35.00
MAPLE	885-8012-[37]	CPM	COMMUNICATION	35.00
MICRONET CONNECTION	885-1122-[37]	HDOS	COMMUNICATION	20.00
MISCELLANEOUS UTILITIES	885-1089-[37]	HDOS	UTILITY	20.00
MORSE CODE TRANSCEIVER	885-8016	HDOS	AMATEUR RADIO	20.00
MORSE CODE TRANSCEIVER	885-8031-[37]	CPM	AMATEUR RADIO	20.00
PAGE EDITOR	885-1079-[37]	HDOS	UTILITY	25.00
PROGRAMS FOR PRINTERS	885-1082	HDOS	UTILITY	20.00
REMARK VOL 1 ISSUES 1-13	885-4001	N/A	1978 TO DECEMBER 1980	20.00
RUNOFF	885-1025	HDOS	TEXT PROCESSOR	35.00
SCICALC	885-8027	HDOS	UTILITY	20.00
SMALL BUSINESS PACKAGE	885-1071-[37]	HDOS	BUSINESS	75.00
SMALL-C COMPILER	885-1134	HDOS	LANGUAGE	30.00
SOFT SECTOR SUPPORT PACKAGE	885-1127-[37]	HDOS	UTILITY	20.00
STUDENT'S STATISTICS PACKAGE	885-8021	HDOS	EDUCATION	20.00
SUBMIT (Z80 ONLY)	885-8006	HDOS	UTILITY	20.00
TERM & HTOC	885-1207-[37]	CPM	COMMUNICATION & UTILITY	20.00
TINY BASIC COMPILER	885-1132-[37]	HDOS	LANGUAGE	25.00
TINY PASCAL	885-1086-[37]	HDOS	LANGUAGE	20.00
UDUMP	885-8004	HDOS	UTILITY	35.00
UTILITIES	885-1212-[37]	CPM	UTILITY	20.00
UTILITIES BY PS	885-1126	HDOS	UTILITY	20.00
VARIETY PACKAGE	885-1135-[37]	HDOS	UTILITY & GAMES	20.00
VOLUME I	885-1008	N/A	SOFTWARE LISTINGS	9.00
VOLUME II	885-1013	N/A	SOFTWARE LISTINGS	12.00
VOLUME III	885-1015	N/A	SOFTWARE LISTINGS	9.00
VOLUME IV	885-1037	N/A	SOFTWARE LISTINGS	12.00
WATZMAN ROM SOURCE & DOC	885-1221-[37]	CPM	H19 FIRMWARE	30.00
WATZMAN ROM	885-4600	N/A	H19 FIRMWARE	45.00
WHEW UTILITIES	885-1120-[37]	HDOS	UTILITY	20.00
XMET ROBOT X-ASSEMBLER	885-1229-[37]	CPM	UTILITY	20.00
Z80 ASSEMBLER	885-1078-[37]	HDOS	UTILITY	25.00
Z80 DEBUGGING TOOL (ALDT)	885-1116	HDOS	UTILITY	20.00
H8 — H/Z-89/90 — H/Z-100 (Not PC)				
ADVENTURE	885-1222-[37]	CPM	GAME	10.00
BASIC-E	885-1215-[37]	CPM	LANGUAGE	20.00
CASSINO GAMES	885-1227-[37]	CPM	GAME	20.00
CHEAPCALC	885-1233-[37]	CPM	SPREADSHEET	20.00
CHECKOFF	885-8011-[37]	CPM	CHECKBOOK SOFTWARE	25.00
COPYDOS	885-1235-37	CPM	UTILITY	20.00
DISK DUMP & EDIT UTILITY	885-1225-[37]	CPM	UTILITY	30.00
DOCUMAT & DOCULIST	885-8019-[37]	CPM	TEXT PROCESSOR	20.00
DUNGEONS & DRAGONS	885-1209-[37]	CPM	GAMES	20.00
FAST ACTION GAMES	885-1228-[37]	CPM	GAME	20.00
FAST EDDY & BIG EDDY	885-8018-[37]	CPM	TEXT PROCESSOR	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 BINDER	885-0004	N/A	REMARK BINDER	5.75
HUG FILE MANAGER & UTILITIES	885-1246-[37]	CPM	UTILITY	20.00
HUG SOFTWARE CATALOG UPDATE #1	885-4501	VARIOUS	PRODUCTS 1983 THRU 1985	9.75
KEYMAP CPM-80	885-1230-[37]	CPM	UTILITY	20.00
MBASIC PAYROLL	885-1218-[37]	CPM	BUSINESS	60.00
MICRONET CONNECTION	885-1224-[37]	CPM	COMMUNICATION	16.00
NAVPROGSEVEN	885-1219-[37]	CPM	FLIGHT UTILITY	20.00
REMARK VOL 3 ISSUES 24-35	885-4003	N/A	1982	20.00
REMARK VOL 4 ISSUES 36-47	885-4004	N/A	1983	20.00
REMARK VOL 5 ISSUES 48-59	885-4005	N/A	1984	25.00
REMARK VOL 6 ISSUES 60-71	885-4006	N/A	1985	25.00
REMARK VOL 7 ISSUES 72-83	885-4007	N/A	1986	25.00
RF CAD	885-8020-[37]	CPM	UTILITY	30.00

PRODUCT NAME	PART NUMBER	OPERATING SYSTEM	DESCRIPTION	PRICE
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 UTILITIES FOR MBASIC	885-1231-[37]	CPM	UTILITY	20.00
ZTERM	885-3003	CPM	COMMUNICATION	20.00
H/Z-100 (Not PC) Only				
ACCOUNTING SYSTEM	885-8048-37	MSDOS	BUSINESS	20.00
CALC	885-8043-37	MSDOS	UTILITY	20.00
CARDCAT	885-3021-37	MSDOS	BUSINESS	20.00
CHEAPCALC	885-3005-37	MSDOS	SPREADSHEET	20.00
CHECKBOOK MANAGER	885-3013-37	MSDOS	BUSINESS	20.00
CP/EMULATOR	885-3007-37	MSDOS	CPM EMULATOR	20.00
DBZ	885-8034-37	MSDOS	DBMS	25.00
ETCHDUMP	885-3005-37	MSDOS	UTILITY	20.00
EZPLOT	885-3023-37	MSDOS	PRINTER PLOTTING UTILITY	20.00
FAST EDDY	885-8025-37	CPM	TEXT PROCESSOR	20.00
FAST EDDY	885-8029-37	MSDOS	TEXT PROCESSOR	20.00
GAMES CONTEST PACKAGE	885-3017-37	MSDOS	GAMES	25.00
GAMES PACKAGE II	885-3044-37	MSDOS	GAMES	25.00
GRAPHICS	885-3031-37	MSDOS	ENTERTAINMENT	20.00
HELPSCREEN	885-3039-37	MSDOS	UTILITY	20.00
HUG BACKGROUND PRINT SPOOLER	885-1247-[37]	CPM	UTILITY	20.00
HUG BACKGROUND PRINT SPOOLER	885-5009-37	CPM86	UTILITY	20.00
HUGPBBS	885-5006-37	CPM86	COMMUNICATION	40.00
HUGPBBS SOURCE LISTING	885-5007-37	CPM86	COMMUNICATION	60.00
ICT 8080 TO 8088 TRANSLATOR & HFM	885-5008-37	CPM86	UTILITY	20.00
KEYMAC	885-3046-37	MSDOS	UTILITY	20.00
KEYMAP	885-3010-37	MSDOS	UTILITY	20.00
KEYMAP	885-5001-37	CPM86	UTILITY	20.00
KEYMAP CPM-85	885-1245-37	CPM	UTILITY	20.00
MAPLE	885-8023-37	CPM	COMMUNICATION	35.00
MATHFLASH	885-8030-37	MSDOS	EDUCATION	20.00
ORBITS	885-8041-37	MSDOS	EDUCATION	25.00
POKER PARTY	885-8042-37	MSDOS	ENTERTAINMENT	20.00
SCICALC	885-8028-37	MSDOS	UTILITY	20.00
SKYVIEWS	885-3015-37	MSDOS	ASTRONOMY UTILITY	20.00
SMALL-C COMPILER	885-3026-37	MSDOS	LANGUAGE	30.00
SPELL5	885-3035-37	MSDOS	SPELLING CHECKER	20.00
SPREADSHEET CONTEST PACKAGE	885-3017-37	MSDOS	VARIOUS SPREADSHEETS	25.00
TERM86 & DSKED	885-5004-37	CPM86	COMMUNICATION & UTILITIES	20.00
TREE-ID	885-3036-37	MSDOS	TREE INDENTIFIER	20.00
USEFUL PROGRAMS I	885-3022-37	MSDOS	UTILITIES	30.00
UTILITIES BY PS	885-5003-37	CPM86	UTILITY	20.00
UTILITIES	885-3008-37	MSDOS	UTILITY	20.00
ZBASIC DUNGEONS & DRAGONS	885-3009-37	MSDOS	GAME	20.00
ZBASIC GRAPHIC GAMES	885-3004-37	MSDOS	GAMES	20.00
ZBASIC GAMES	885-3011-37	MSDOS	GAMES	20.00
ZPC II	885-3037-37	MSDOS	PC EMULATOR	60.00
ZPC UPGRADE DISK	885-3042-37	MSDOS	UTILITY	20.00
H/Z-100 — PC Compatibles				
ADVENTURE	885-3016-37	MSDOS	GAME	10.00
ASSEMBLY LANGUAGE UTILITIES	885-8046-37	MSDOS	UTILITY	20.00
DEBUG SUPPORT UTILITIES	885-3038-37	MSDOS	UTILITY	20.00
DOCUMAT & DOCULIST	885-8035-37	MSDOS	TEXT PROCESSOR	20.00
DPATH	885-8039-37	MSDOS	UTILITY	20.00
HADES	885-3040-37	MSDOS	UTILITY	40.00
HELP	885-8040-37	MSDOS	CAI	20.00
HEPCAT	885-3045-37	MSDOS	UTILITY	35.00
HUG BACKGROUND PRINT SPOOLER	885-3029-37	MSDOS	UTILITY	20.00
HUG BINDER	885-0004	N/A	REMARK BINDER	5.75
HUG EDITOR	885-3012-37	MSDOS	TEXT PROCESSOR	20.00
HUG MENU SYSTEM	885-3020-37	MSDOS	UTILITY	20.00
HUG SOFTWARE CATALOG UPDATE #1	885-4501	VARIOUS	PROD 1983 THRU 1985	9.75
HUGMCP	885-3033-37	MSDOS	COMMUNICATION	40.00
HUGPBBS SOURCE LISTING	885-3028-37	MSDOS	COMMUNICATION	60.00
HUGPBBS	885-3027-37	MSDOS	COMMUNICATION	40.00
ICT 8080 TO 8088 TRANSLATOR	885-3024-37	MSDOS	UTILITY	20.00
MATT	885-8045-37	MSDOS	MATRIX UTILITY	20.00
MISCELLANEOUS UTILITIES	885-3025-37	MSDOS	UTILITIES	20.00
REMARK VOL 5 ISSUES 48-59	885-4005	N/A	1984	25.00
REMARK VOL 6 ISSUES 60-71	885-4006	N/A	1985	25.00
REMARK VOL 7 ISSUES 72-83	885-4007	N/A	1986	25.00
SCREEN DUMP	885-3043-37	MSDOS	UTILITY	30.00
UTILITIES II	885-3014-37	MSDOS	UTILITY	20.00
PC Compatibles				
ACCOUNTING SYSTEM	885-8049-37	MSDOS	BUSINESS	20.00
CARDCAT	885-6006-37	MSDOS	CATALOGING SYSTEM	20.00
CHEAPCALC	885-6004-37	MSDOS	SPREADSHEET	20.00
CP/EMULATOR II & ZEMULATOR	885-6002-37	MSDOS	CPM & Z100 EMULATORS	20.00
DUNGEONS & DRAGONS	885-6007-37	MSDOS	GAME	20.00
EZPLOT	885-6003-37	MSDOS	PRINTER PLOTTING UTILITY	20.00
FAST EDIT	885-8033-37	MSDOS	TEXT PROCESSOR	20.00
GRADE	885-8037-37	MSDOS	GRADE BOOK	20.00
HAM HELP	885-6010-37	MSDOS	AMATEUR RADIO	20.00
KEYMAP	885-6001-37	MSDOS	UTILITY	20.00
LASERWRITER CONNECTION	885-8050-37	MSDOS	PRINTER UTILITY	40.00
RF CAD	885-8038-37	MSDOS	UTILITY	30.00
SCREEN SAVER PLUS	885-6009-37	MSDOS	UTILITIES	20.00
SKYVIEWS	885-6005-37	MSDOS	ASTRONOMY UTILITY	20.00
TCSPELL	885-8044-37	MSDOS	SPELLING CHECKER	20.00

You've got a screen full of important technical data that would be nearly impossible to memorize, and you already have writer's cramps from the last screen full. With ***SCREENDUMP*** from HUG, you can reproduce a complete video screen on a dot matrix printer, including both text and graphics without having to exit the current program. *SCREENDUMP* supports most of the more popular dot matrix printers, including the newer 24-pin and laser jet models. The latest version of *SCREENDUMP* is ***HUG P/N 885-3043-37***.

"Thank Heaven for *HADES*!" That's what a lot of MS-DOS users are saying when ***HADES*** rescues a file that just got accidentally erased. Erased file recovery is only a small part of the capabilities of this program. *HADES* is *HUG's Absolute Disk Editing System*. Within the realms of MS-DOS, *HADES* allows you to directly edit any part of any disk. Directories, files, file attributes. FATS: nothing can hide from you when you use *HADES*. *HADES* works on *ANY* computer that can run MS-DOS version 2 or greater. Order ***HUG P/N 885-3040-37*** today!

Want to keep your H/Z-100? Want to run a lot of that good PC compatible software out there? Don't want to buy a PC compatible though? Then get ***ZPC II, HUG P/N 885-3037-37***, and the ***ZPC II upgrade disk, HUG P/N 885-3042-37***.

ORDERING INFORMATION

For VISA and MasterCard phone orders, telephone the Heath Users' Group directly at (616) 982-3838. Have the part number(s), descriptions, and quantity ready for quick processing. By mail, send your order, plus 10% postage and handling ($1.00 minimum charge, up to a maximum of $5.00) to: Heath Users' Group, P.O. Box 217, Benton Harbor, MI 49022-0217. VISA and MasterCard require minimum $10.00 order. No C.O.D.s accepted.

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


"Why Haven't You Called?"
"I've stayed up 24 hours a day waiting to hear from you. I know that for most of you, this may be your first time, but I promise, I'm very gentle! If you're still unsure, let me tell you what I have to offer. First, there's my message base. Here's where my users exchange ideas, receive assistance, and sell things. Next, is my database. Most of my users will tell you, it's second to none! With over 30 megabytes of hand-picked software it caters to all Heath/Zenith computers, but mostly PC compatibles. Finally, I have something new for you, my 'Bargain Centre'. Here you can buy surplus software and hardware, at unheard of prices. Interested? I hope so! Set your modem to either 300, 1200, or 2400 baud, and call me right now at (616) 982-3956, and register today. If you're still a bit shy, you can still call my human at (616) 982-3837 and register with him. Although he talks at 150 baud, he's gentle, too!"
MoC

ZPC Update #20

Pat Swayne
HUG Software Engineer

This is the twentieth in a series of articles in support of ZPC, a program that allows you to run IBM PC software in H/Z-100 (dual processor) computers. ZPC is available from HUG as part no. 885-3037-37. An upgrade disk for ZPC is also available as part no. 885-3042-37.

In this update, I will present a patch for Logitech Publisher (PFS: First Publisher), LOGICADD version 2.02, and Multiplan version 3.0 to allow them to run under ZPC. I will also give you some instructions for running Turbo C under ZPC (no patch required).

Logitech Publisher Patch

Logitech Publisher is the same thing as PFS: First Publisher packaged with a logitech mouse, but it is not known if the patch will work with the version sold by PFS. A DEBUG listing of the patch area will be shown to help you patch your version if the patch presented here does not work.

To patch Logitech Publisher to run under ZPC, add the following lines to your PATCHER.DAT file:

```
LOGITECH PUBLISHER (PFS) v 1.0
Insert the disk containing FP.EXE and
  SNAPSHOT.COM.
FP.EXE
B3DA,90
B3E4,90
B3EB,B4,03,CD,10,89,C8,90
x
SNAPSHOT.COM
98C,CD,90
9B0,CD,91
F9F,64
FA6,40
z
```

As you can see, the SNAPSHOT program is also patched for use with ZPC. It will work, but only if you specify an activation key sequence other than Shift-Prt Sc. A good sequence to use is Alt-B. When you want to load SNAPSHOT, enter SNAPSHOT K at the DOS prompt, and when it asks for a key, press the Help key followed by the B key (the ZPC way to enter Alt-B), and use that sequence again whenever you want to take a snapshot of the screen. Note that some key sequences may not work, and some will interfere with Logitech Publisher, but Alt-B has been tried, and it works.

Since I had access to the Logitech Publisher program only and not a Logitech mouse, I do not know in what way, if any, the mouse can be supported under ZPC. However, the program is very easy to use without the mouse, and screen response would probably be too slow under ZPC to keep up with one if you had one.

If the patch listed here does not work with your version of Logitech Publisher or PFS: First Publisher, use DEBUG to look for code like this:

```
2D97:B4D5 BAD403        MOV     DX,03D4
2D97:B4D8 B00E          MOV     AL,0E
2D97:B4DA EE            OUT     DX,AL
2D97:B4DB 42            INC     DX
2D97:B4DC EC            IN      AL,DX
2D97:B4DD 8AD8          MOV     BL,AL
2D97:B4DF B90800        MOV     CX,0008
2D97:B4E2 8AC1          MOV     AL,CL
2D97:B4E4 EE            OUT     DX,AL
2D97:B4E5 A17D05        MOV     AX,[057D]
2D97:B4E8 EC            IN      AL,DX
2D97:B4E9 3AC1          CMP     AL,CL
2D97:B4EB 7507          JNZ     B4F4
2D97:B4ED E2F3          LOOP    B4E2
2D97:B4EF 8AC3          MOV     AL,BL
2D97:B4F1 EE            OUT     DX,AL
2D97:B4F2 3BC0          CMP     AX,AX
2D97:B4F4 C3            RET
```

Modify the code so that it looks like this:

```
2D97:B4D5 BAD403        MOV     DX,03D4
2D97:B4D8 B00E          MOV     AL,0E
2D97:B4DA 90            NOP
2D97:B4DB 42            INC     DX
2D97:B4DC EC            IN      AL,DX
2D97:B4DD 8AD8          MOV     BL,AL
2D97:B4DF B90800        MOV     CX,0008
2D97:B4E2 8AC1          MOV     AL,CL
2D97:B4E4 90            NOP
2D97:B4E5 A17D05        MOV     AX,[057D]
2D97:B4E8 EC            IN      AL,DX
2D97:B4E9 3AC1          CMP     AL,CL
2D97:B4EB B403          MOV     AH,03
2D97:B4ED CD10          INT     10
2D97:B4EF 89C8          MOV     AX,CX
2D97:B4F1 90            NOP
2D97:B4F2 3BC0          CMP     AX,AX
2D97:B4F4 C3            RET
```

LOGICADD Patch

The following patch for LOGICADD version 2.02 was provided by Janet Hirsch of

Hogware Company. LOGICADD is the Logitech version of Generic Cadd, and patches for other versions of LOGICADD and Generic Cadd have appeared in previous issues of REMark. Here are the patcher lines for this version:

```
LOGICADD version 2.02 12-02-86
Insert the disk containing CADD.EXE.
CADD.EXE
14432,90
14458,90
1445D,90
1447D,90
144A9,90
144AE,90
145E9,90
14694,90
146EC,B0
14756,B0
14793,B0
147FC,B0
1484D,0,0
148B5,0,0
14930,0,0
z
LOGICADD DOTPLOT 7-17-86
Insert the disk containing DPLOT.EXE.
DPLOT.EXE
5A13,B0
5A7D,B0
5ABA,B0
5B23,B0
5B74,0,0
5BDC,0,0
5C57,0,0
z
```

Multiplan 3.0 Patch

Multiplan version 3.0 can be run under ZPC in mode 7 after a simple patch has been performed. If you want to run it in color, a much more complicated patch is required. If you only want to use mode 7, use the following lines for your patch:

```
MULTIPLAN v 3.0 in mode 7
Insert the disk containing MP.COM
MP.COM
25F2,CD,90,08,C0
z
```

If you want to run Multiplan in color (it will still work in mode 7 with this patch), use these lines for your patch:

```
MULTIPLAN v 3.0 in color
Insert the disk containing MP.COM
MP.COM
25F2,CD,90,08,C0
27BA,90
27BD,90
27C4,90
27C7,90
281F,90
2885,90,90,90,90,90,90,90,90,90,90
2D3C,90,90,90,90,90,90,90,90,90,90
2D7B,90,90,90,90,90,90,90,90,90,90
2E67,90,90,90,90,90,90,90,90,90,90
2E91,90,90,90,90,90,90,90,90,90,90
2EC0,90,90,90,90,90,90,90,90,90,90
2F7B,90
2F8D,90
2FA6,90
2FB8,90
2FF1,90
2FF9,90
301E,90,90,90,90,90,90,90,90,90,90
436A,90
436D,90
4374,90
4377,90
43CF,90
4435,90,90,90,90,90,90,90,90,90,90
48EC,90,90,90,90,90,90,90,90,90,90
492B,90,90,90,90,90,90,90,90,90,90
4A17,90,90,90,90,90,90,90,90,90,90
4A41,90,90,90,90,90,90,90,90,90,90
4A70,90,90,90,90,90,90,90,90,90,90
4B2B,90
4B3D,90
4B56,90
4B68,90
4BA1,90
4BA9,90
4BCE,90,90,90,90,90,90,90,90,90,90
z
```

Multiplan 3.0 will run under mode 3 after you make this patch, and you can use the Window Paint command to change the colors to your liking.

Make the patches presented in this article using PATCHER and the instructions in your ZPC manual. All of the programs mentioned above will require patching to run under ZPC regardless of whether you have a Scottie board or not.

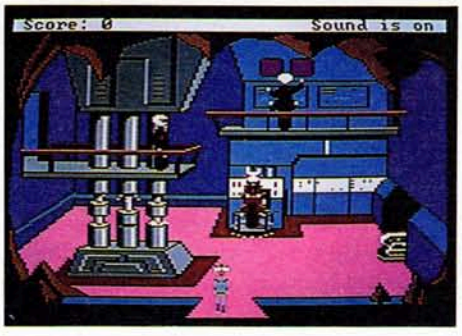
Running Turbo C

Two ZPC users, Richard Ferch and John Collins, sent in information on running Turbo C under ZPC. I have not tried it myself, so this report is based on their information.

Only the integrated development version of the Turbo C program requires any preparation. The other programs supplied, including the command line version of the compiler, will run as is. To prepare the integrated development version of the compiler (TC.EXE), put ZPC into video mode 7 and run TCINST. Type S to get to the screen installation menu. Select the default video mode, and when the program runs a "snow check", indicate that there is no snow. Finish running the TCINST according to instructions, and then you should be able to run TC.EXE under ZPC. *

All Work And No Play . . . Or "Trust Me Luke, Use The Farce . . ."

Jim Buszkiewicz
HUG Managing Editor



Space Quest

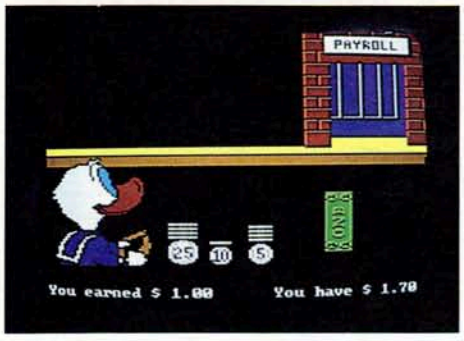
Far beyond the reaches of known space lies a planetary system known to its inhabitants as Earnon. For many years, Earnon's sun has been slowly dying. The only hope for Earnon's survival has rested in the scientific community on the governing planet of Xenon, which has been working for quite some time on a plan to convert one of the system's lifeless planets into a new sun. The effort has been centered around the development of a device called the Star Generator. The development team for the project has been stationed aboard the spacelab Arcada, and sent to the outer edge of the Earnon to further its research. You, a lowly apprentice sanitation engineer, are employed on the Arcada in tasks requiring only the lowest security clearance. It is common knowledge aboard the Arcada that the scientists have just successfully completed the development and first stages of testing of the Star Generator. Although still in the experimental stages, it appears to be fully operational. Triumphantly, the good news is flashed back to Xenon as the crew of the Arcada prepares for the trip home. The news, however, does not travel far before it reaches unintended ears. Monitoring the Arcada's transmissions, are Sariens, space pirates who cruise the galaxies wreaking havoc. Their immediate intent is to capture the Star Generator and bring it aboard their battle cruiser. The attack comes. As your adventure begins, you are currently conducting one of your famous on-shift naps in the janitorial storage closet. Suddenly, you are rudely awakened by explosions that rock the ship. Immediately, the alarm system is activated and an announcement comes over the intercom that Sarien troops have boarded the ship. What will you do?!

This is the main plot for Sierra's 3-D animated adventure game, "Space Quest, The Sarien Encounter". Using the arrow keys on your PC's keyboard, you control the movement of your character through the various scenes presented throughout the game. The scenery graphics, and background animation, add to the realism of the game, and makes you feel as if you're actually 'there'. Your character can perform special actions such as examining items, pushing buttons, by typing in the actual command at the keyboard.

I thought my game playing days were over until I tried Space Quest. The only real adventure game I ever completed was the original "Adventure" by Crowther and Woods. Other variations could never hold my interest. I feel the reason for this is that each puzzle in the adventure required an exact solution which kept you guessing and madly typing until you just happened upon the answer (a more complex variation of the old 'number guess' game). Space Quest is different. It's logical and life-like. In most cases, you know what needs to be done, it's finding the resources to do it that becomes the puzzle, and in some instances, the puzzle can be solved in more than one way.

Your character will get involved with four different major scenarios. The first will be on the spacelab Arcada. Here, he'll have to deal with the invasion force, gather clues, and escape. Next, you'll stop off at a planet where your character will visit a subterranean world of a super intelligent race. The question is, though, are they friendly? A small settlement on the planet of Kerona is his next stop. There your character will mingle with undesirable aliens. Finally, if you prove resourceful enough to make it this far, you'll venture deep into the bowels of the Sarien battle cruiser, "Deltaur". Here,

you will come face to face with those bad boys of the universe. And, if you actually finish your quest, you'll be rewarded with a scene that is reminicent of the final scenes from "Star Wars". I can't even begin to describe the exhilarating feeling I had when I finished MY quest! Now I can hardly wait for "Space Quest II", the sequel, to become available.



Donald Duck's Playground

In the past, very little has been available for grade school children in the form of games that helped teach. Either the games were strictly tutors, or mindless shoot-em-ups. Sierra On-Line has changed that with their release of "Donald Duck's Playground". This game teaches youngsters (ages 7 to 11) how to count money, make change, pay for purchases, and keep track of their assets.

Donald wants to build a playground for his nephews, Huey, Dewey and Louie, but lacks the money to purchase the equipment. To come up with the cash, Donald takes on vaious jobs. These jobs include, working at a produce stand, a toy store, an airline depot, and a railroad. He can work at as many of the different jobs as he likes, for as long as he likes in order to earn the amount of cash he needs.

Being a kid at heart, Donald's first choice is the Toy Store. Here Donald must stow the new merchandise on the shelves. Each new toy first appears on the loading dock, and the game has to match the item on the storage shelf. Using the ladder, Donald picks up the new toy, and places it beside the matching item. This process continues until it's time for the Amquack Special to rumble past. When this occurs, the toys tend to slide off the shelves and break. Keeping a close eye on the train schedule, Donald must close the toy cabinet doors before each train comes, and then reopen them after it goes past. Each toy correctly placed on the shelves, earns Donald some money. If, however, a toy should fall, its value is deducted from Donald's pay.

Another job Donald can work at, is the fruit and vegetable market. Here he sorts produce into bins. Melons, pumpkins, fruit, and veggies, are tossed from the back of a produce truck for Donald to catch and drop into their proper bins.

The third job Donald handles is at the McDuck Airlines. Here he sorts packages into cargo carts. The screen displays a plane landing, then a train of cargo cars driving to where a conveyor belt of packages waits to be loaded. Donald must throw each box into the correct tram, matching the three-letter destination codes on the packages to those on the tram cars. Donald earns money for each package correctly loaded.

The most interesting place for Donald to work at is the Amquack Railroad, where he's the dispatcher and switchman. The miniature railroad layout can be seen from overhead, and as the train picks up packages for delivery, Donald must open or close junction switches so the train is routed to the correct towns.

Once Donald has earned some money, he can then walk to the other side of town where the shops are located. At these shops, he can buy various pieces of playground equipment, using the money he's earned. Once Donald choose something to buy, the scene changes to a cash register screen. By moving a cursor to select the bills and coins needed to cover the purchases and making change as necessary, Donald buys his new equipment. Donald's selections are automatically delivered and installed at the playground at the end of Main Street. Donald can visit the playground at any time to see what else may be needed. To get to the playground, Donald must walk down the street and cross the tracks. Before doing so, he does look both ways before crossing.

The playground sequence features one of Donald's nephews, animated by the player. He can make the duck swing, climb ladders and nets, slide, and bounce on a toy horse.

This game is undoubtedly one of the best educational games available. It's specifically designed to teach kids to count money and make change. The best part is, they'll be having loads of fun while doing it! Unlike most Sierra products, this game does not appear to be copy-protected.

Helicopter Simulator

This game, (which is not really a game at all, but a simulation) is the newest product released by Sierra On-Line Inc. This product simulates, in 3-D with excellent graphics, you behind the controls of a helicopter. In direct competition with MicroSoft's "Flight Simulator", Helicopter surpasses the former's capabilities in many ways.

Some of the features include flight maneuvers that aren't possible with airplane or jet simulators. You can fly sidewards, backwards, or rotate a full 360 degrees while hovering in mid-air. The scenery is fully animated, and special effects are included, such as fast or slow scan, to give you a full 360 degree view of your surroundings. This includes a zoom control for extreme close-ups. Also, you have the choice of incredible out-of-ship points of view! You can overview your helicopter's flight from the radar tower, or admire your ship from a ground level point-of-view. You can even examine your flight manuevers from a tracking plane that is following directly behind your ship! Finally, there are four playing modes to choose from: Flight, Target Practice, Combat (against the computer), and Optional Two-Player Combat (through modem hook-up)!

This last feature (Two-Player Combat), allows you to go head-to-head with another player (running the same program). The two computers must be connected either via modem and phone line at 1200 baud, or directly via null modem cable.

I found Helicopter somewhat difficult to operate (after all, this was a real simulation). I could initially manuever in the air, but couldn't land. The manual indicated that this would be tricky, but possible. I didn't think there should be anything tricky about approaching the ground very slowly in a vertical direction. The simulation, however, wouldn't let me land! Every time I touched down slowly, it popped me back in the air. If I touched down slightly too fast, I crashed. Unless you're really interested in flying, I found Helicopter frustrating and boring, but that's just me.

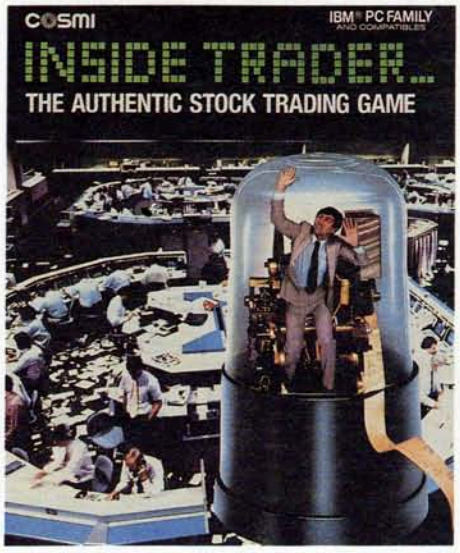
Sierra Copy Protection

Most of Sierra On-Line's products are copy protected. When initially executed, you must have the original disk in drive A: for the program to continue execution. This copy protection scheme can easily be surpassed using the NOKEY program from Central Point Software. This program comes on their COPYIIPC disk.

Inside Trader

Inside Trader is a game produced by COSMI CORP., that provides a realistic simulation of insider trading on the stock

market. The player can make stock purchase decisions based on inside information to increase the value of his portfolio. The object is to avoid getting caught by the Securities and Exchange Commission.



Based upon actual events, the player receives market quotations (on his printer), news events (on his CRT), prospectuses, "inside" and random information as he plans his strategy and executes his orders. By using discretion and daring, the player can make millions. But, if he gets greedy or foolish, he may end up in the "slammer"!

"Hot keys", using the function keys on the PC keyboard make market transactions lightning fast and easy. Plus, this exciting action strategy game can be used with actual stocks traded on any exchange. This program also supports all display types, including MDA, CGA, EGA, and HERCULES.

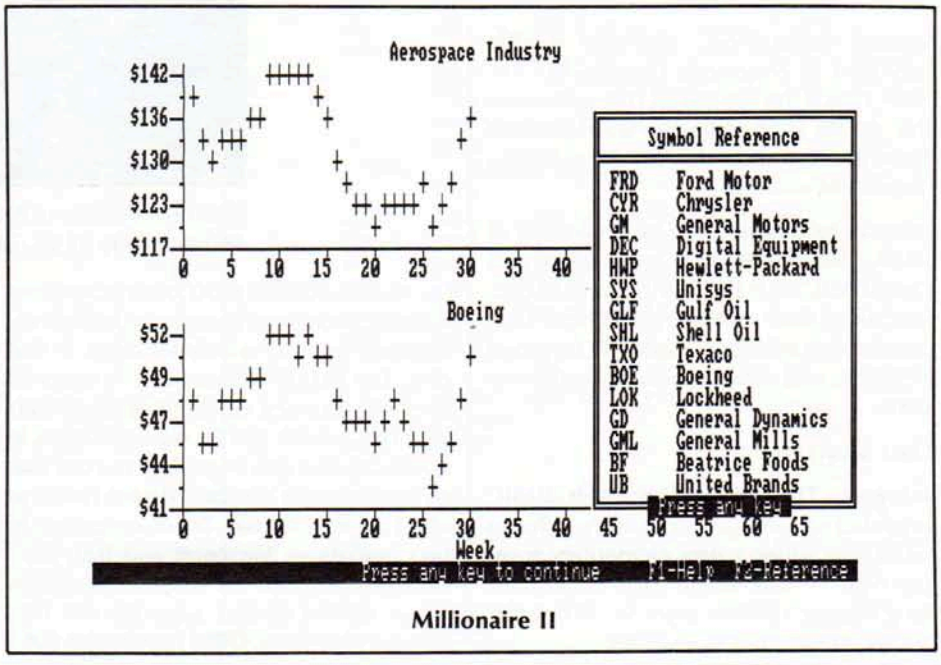
I'm not a business-type person. I even hate signing my name to checks, but this game was fun! The variety of ficticious companies and events were hilarious, but could have represented actual happenings. I didn't even mind losing thirty some odd thousand dollars on my first try! This game ran in real time, was easy to learn and play. I could even recommend it for junior high school age kids, and it's not copy protected!

Millionaire II

This game is an adventure into the world of stock market investing. The original game of Millionaire was reviewed in the August 1985 issue of REMark. This version appears to be quite similar, although not as luxuriously packaged. Again, I would not call this product a game. It's actually a complex real-life simulation! Also included with the product is a short tutorial on the American Stock Exchange and how it all works. Simply put, the object is to buy and sell stocks until you become a millionaire. This version is menu driven, with help available at most prompts. Millionaire II is copy-protected, but is easily copied using Central Point Software's COPYIIPC program.

Products Reviewed

Space Quest	$49.95
Donald Duck's Playground	$24.95
Helicopter Simulator	$39.95
Sierra On-Line Inc. P.O. Box 485 Coarsegold, CA 93614 (209) 683-6858	
Inside Trader	N/A
Cosmi 415 N. Figueroa Wilmington, CA 90744 (213) 835-9687	
Millionaire II	$59.95
Blue Chip software Dept. 687 185 Berry Street San Francisco, CA 94107	
COPYIIPC & NOKEY	$39.95
Central Point Software Inc. 9700 SW Capitol Hwy. #100 Portland, OR 97219 (503) 244-5782	*



Millionaire II

BASIC Wars

Bruce Dubbs
122 Valencia Drive
Universal City, TX 78148

A long time ago, in a galaxy far away, there was Microsoft Interpreted BASIC. Today there is a battle. It is the BASIC battle for the hearts and minds of BASIC programmers everywhere. The original shot was made when QuickBASIC 1.0 was released by Microsoft. Then the people from Borland expressed an intention to invade by producing Turbo BASIC. Microsoft retaliated with QuickBASIC 2.0. Then Borland struck with delivery in April 1987 of the eagerly awaited Turbo BASIC. Microsoft quickly retaliated in May with QuickBASIC 3.0. Now, how is the war going? The purpose of this article is to compare and contrast Turbo BASIC and QuickBASIC 3.0 to let you decide the winner.

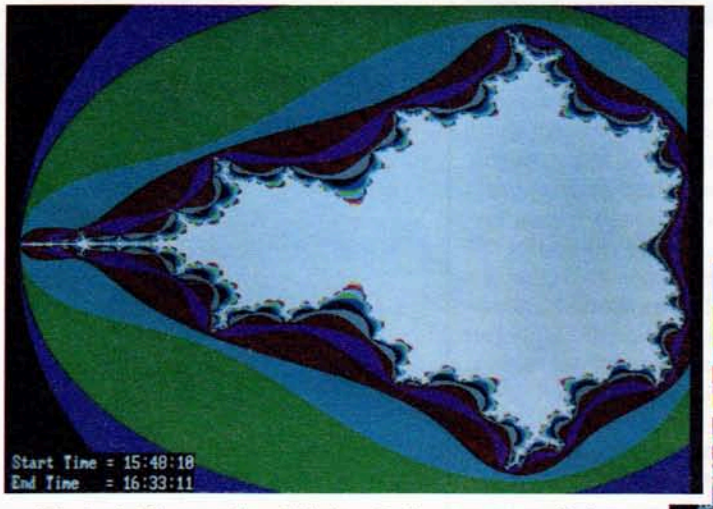


Photo 1: The results of Listing 5 show an overall view of the Mandelbrot Set.

In order to describe the new versions of BASIC, I will discuss how Turbo BASIC and QuickBASIC each address the areas of data structures, flow control, speed and size, programmer interface, assembly language interface, and new commands and functions.

Data Structures

Integers. The original Microsoft BASIC provided one integer number type. Turbo BASIC has added a new elementary number type — long integer. The suffix for a long integer variable is an '&'. What this allows you to do is to execute:

```
BIG.NUMBER& = 30000 + 30000 + 30000
```

QuickBASIC does not support long integers.

Floating Point. When considering numbers, it is often important for a programmer to have an understanding of what the internal structure of a number is doing. For example, if you tried

```
SUM = 100000.0 + 0.00001
```

the result would be 100000.0, because the internal precision of a single precision floating point number is only six digits. In this case, the .00001 is truncated. In order to preserve accuracy, double precision floating point, which has 15 digit precision, is necessary. It is also important to note that all floating point numbers do not have the same internal format. There are generally two formats — Microsoft and IEEE standard. QuickBASIC 3.0 includes a version called QB87.EXE that supports the IEEE standard numbers. QB87 requires an 8087 or 80287 math coprocessor to work. The standard QuickBASIC uses the Microsoft format found in interpreted BASIC. Turbo BASIC does all its internal calculations in the IEEE standard format. Both Turbo BASIC and QuickBASIC provide functions to convert numbers to and from the Microsoft and IEEE standard formats.

Strings. The string data type in QuickBASIC and Turbo BASIC are the same. They differ from interpreted BASIC in that they support strings of up to 32,767 characters. Interpreted BASIC only supported strings of 255 characters. Both QB3 and TB have a limit of 64 Kbytes of total string space.

There is a difference, however. TB uses all of its 64 Kbytes for strings. QB3 uses its block to hold strings and their descriptors. This reduces the amount of true string space that QB3 can handle.

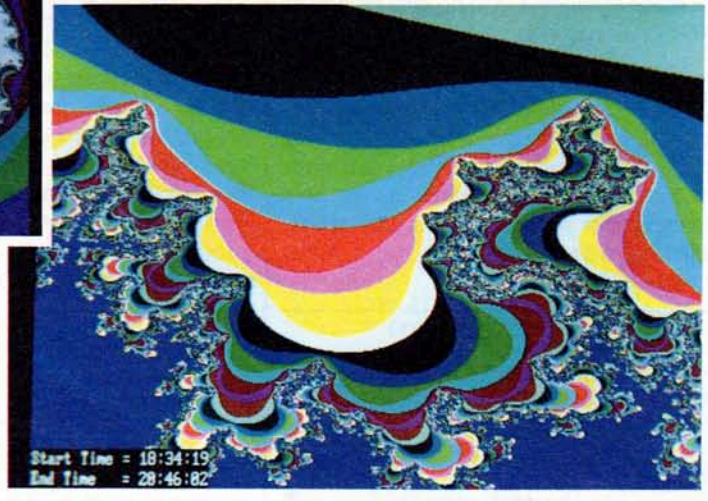


Photo 2: Results of Mandelbrot algorithm at location -1.0, .35, and a density of 48. The entire window is .1 high and .1 wide.

Variables. Variable names are much more flexible in QB3 and TB than they were in Interpreted BASIC. Because both are compilers, the length of a variable name makes no difference in execution speed. This allows names, such as "EndOfYear.Total" to be practical without an execution speed penalty. QuickBASIC limits variable names to 40 characters. TB has no limit (I quit at 60 characters when I tried it.) The QB3 limitation is not really significant.

Arrays. With both TB and QB3, data space for arrays is much larger than under Interpreted BASIC. Both allow EACH array to be as large as 64 Kbytes. Arrays also can be dynamic or static. Static arrays are always present within a program, and only have to be initialized once. Dynamic arrays can be allocated at execution time and can be redimensioned as the program needs are encountered. A big plus with Turbo BASIC is the ability to declare the bounds of an array. For example, an array can be defined as

```
INCOME(1980:1989)
```

to hold data representing years 1980 to 1990. In this case, the array only uses 40 bytes of memory, because it represents 10 single precision floating point numbers.

Named Constants. TB and QB3 have introduced the concept of named constants to BASIC. Turbo BASIC limits named constants to integers. QuickBASIC allows named constants of any data type and indeed allows named constants to be made up by simple expressions. This allows names to be used in many places where a number is now used, enhancing documentation of the program.

Flow Control

Flow Structures. Like Interpreted BASIC, TB and QB3 have the WHILE...WEND and FOR...NEXT statements for loop control. They also have a new statement, DO... LOOP. This new statement is similar to the WHILE...WEND statement, however, the test for completion is made at the end of the loop, guaranteeing execution of the loop at least once.

New flow control statements available in QB3 and TB are the IF block statement and the SELECT CASE statement. The IF block statement allows multiple tests over multiple lines. The SELECT CASE statement allows testing of multiple cases and is a convenient replacement for a long IF... THEN...ELSEIF...ENDIF combination.

Labels and GOTOs. Unlike Interpreted BASIC, labels are optional in QB3 and TB. In addition, labels may be alphanumeric, instead of strictly numeric. These features allow much better structuring of code than was possible before. A restriction in Turbo BASIC is that alphanumeric labels must be on a line by itself. QB3 does not have this limitation.

At this point, a comment is necessary about the GOTO controversy. Almost everyone who has written medium to large programs will tell you that GOTOs are bad. College professors often insist that programs be written with no GOTO statements. The problem with GOTOs is that it is difficult to understand the flow of a program that makes heavy use of GOTOs. Debugging is difficult, as well as program modification. As you trace through a program with a lot of GOTOs, a major problem arises in that it is difficult to tell where the program flow came from without scanning the entire program. GOTOs also tell where program flow diverges, but does not tell you where the program structure rejoins. Why, then, does almost every programming language include the GOTO as a valid command? For one thing, the GOTO is the simplest form of flow control. It can be used to simulate many other flow structures. Moreover, the judicious use of a GOTO or two can also make some otherwise difficult control structure with multiple layers of nesting simpler. The bottom line is that other flow structures are usually appropriate, but a few well placed GOTOs may be just the thing to enhance your program.

Listing 1
Print Numbers

```
DEFINT A-Z

START$ = TIME$
FOR I = 1 TO 32000
  LOCATE 10,10
  PRINT I
NEXT I
LOCATE 20,1
PRINT "Finish Time = " TIME$
PRINT "Start Time  = " START$
END
```

Functions. There are two new features in user defined functions under both QB3 and TB. You can now define multi-line functions and you can declare variables within a function to be local. The significance of local variables is great. This will allow simple variables to be used without concern that something will change in the main program. For instance, the variable "I" is commonly used as a counter. Declaring "I" as local will allow the name to be used in your function, without changing a variable in your main program. Note that the default for variables in functions is global, so local variables have to be explicitly defined.

Subroutines. The most important change in flow control in TB and QB3 is subroutines. Both still support the GOSUB... RETURN combination without changes from Interpreted BASIC, but both also add the capability for true subroutines. The syntax for calling a subroutine is CALL subname [(parms)]. The subroutine is defined with a SUB subname [(parms)] ... END SUB combination. A RETURN statement is not used. The value of subroutines is that all variables default to being local to a subroutine. When this happens, many cases of inadvertent changing of global variables are eliminated. It also allows names to be used instead of line numbers. These features add a new and powerful dimension to BASIC and allow libraries to be developed that cannot accidentally interfere with other code.

Listing 2
Print A DOT

```
DEFINT A-Z

SCREEN 9
START$ = TIME$
FOR I = 1 TO 32000
  PSET (100,100),5
NEXT I
LOCATE 20,1
PRINT "Finish Time = " TIME$
PRINT "Start Time  = " START$
END
```

As a bonus, in Turbo BASIC, subroutines can be recursive. That is, a subroutine can call itself. This capability is at once dangerous and powerful. An incorrectly coded recursive subroutine can run forever (or until it runs out of memory), but a correctly coded routine can solve complicated problems in a few lines of elegant code.

Speed And Size

Trying to evaluate speed and size for a compiler is a tricky business. Everybody has their own favorite benchmarks to decide what program or system runs faster than another. The benchmarks I selected test number crunching and screen output. I elected not to test file I/O primarily be-

Function	TB wo/87	QB	TB w/87	QB87
Print Numbers	249	83	193	96
Draw a dot	21	7	6	5
Draw random dots	93	27	17	24
Draw Boxes	186	85	73	80

Table 1
Relative Execution Speeds - Seconds

cause of the dependency such as test has on disk hardware. In the tests, I used an 8 MHz Z-248 running with one wait state and doing screen output to a Zenith EGA card.

Most tests were performed both with and without an 80287 coprocessor. The listings for the tests are shown in Listings 1 through 5. The results of Listings 1 through 4 are in Table 1. It is interesting to note here that for Listing 1, QB is slower with an 80287 than it is without it. Overall, the increase in speed for these calculations with the 80287 is quite good, and the accuracy is much improved.

The results of Listing 5 are shown in Table 2 for several different options for compilation. In all cases, programs were compiled to optimize speed. As a side issue, Listing 5 is an example of a mathematical function called the Mandelbrot Set. For a more complete discussion of this interesting function, see the accompanying sidebar. Listing 5 is much more numerically intensive than Listings 1 through 4, and the timing results clearly show the advantage of the math coprocessor.

Listing 3
Print DOTs

```
DEFINT A-Z

SCREEN 9
START$ = TIME$
FOR I = 1 TO 32000
  PSET (639*RND,349*RND),15*RND
NEXT I
LOCATE 20,1
PRINT "Finish Time = " TIME$
PRINT "Start Time  = " START$
END
```

Other issues in evaluating a compiler are how fast does the compiler work and what is the size of the code generated. To test this, a single program of 579 lines of BASIC code was compiled using both TB and QB3. The results are shown in Table 3. Note that the size for QB3 is shown in two different formats. The first is when the program is compiled to an "EXE" program. The second is when the program is compiled and linked to use the QuickBASIC run-time module. This entry is shown as code size, plus run-time module size.

Upon examining the results in the Tables, it is evident that the internal optimization of the number formats of QuickBASIC allow faster execution, but there is a size penalty. It is also evident that a math coprocessor makes a big difference for both compilers and that the speed difference between compilers with the coprocessor is not always as significant as it is without the 80287.

	Time	Size
TB w/80287	17:44	23,264
TB wo/80287	121:05	30,704
QB	45:01	45,732 3,857 + 70,680
QB87	14:27	41,058 3,936 + 76,112
QB w/debug	47:15	47,730 4,240 + 70,680

Table 2
Relative Speed (Minutes) and Size (Bytes) of Listing 5 under various compilation options.

Programmer Interface

An extremely important factor in any computer program is how that program relates to the user. Both TB and QB3 provide a complete program development environment to the programmer. Both provide a fast built-in editor, pull-down menus, and built-in debugging aids. A major difference between QB3 and TB is that QB3 can compile to an external object code, ready for linking with other compiled (Fortran, C, BASIC, or Assembly) modules. TB cannot.

Error detection. TB and QB3 have quite different philosophies when it comes to error checking. TB stops immediately when it finds any error. Because of this, you may have to run the compiler a LOT of times to get a medium to large program to compile properly. QB3 on the other hand tries to find all the errors in the program. It is true that many "errors" are flagged because of previous errors, however, QB3's overall approach allows finding and correcting errors quick and easy.

Commands. The command level of the two compilers are very similar. The primary difference is that QB3 is set up for optimal use with a mouse. The keystrokes required by QB3 usually start with the ALT key. For instance, to exit QB3, the programmer must press ALT-F, then ALT-Q. Alternatively, he can press ALT-F, then Q, then RETURN. TB uses the same keys, but without the ALT key. The difference is that QB3 is always in edit mode unless a pull-down window is present. With TB, you must enter edit mode with an E and leave with an ESC. QB3 is much easier to use with a mouse.

Editor. The built-in editors of QB3 and TB are quite different. TB's editor is completely customizable and has a rather slick installation program to allow the customization. QB3's editor is not customizable and is designed for optimal use with a mouse. Since I do not have a mouse, I found it easier to do major editing of a program in a separate text editor, and only minor updates and debugging from within the QB3 editor.

Listing 4
Print Boxes

```
DEFINT A-Z

SCREEN 9
START$ = TIME$
FOR I = 1 TO 32000
  X = 629*RND
  Y = 339*RND
  LINE (X,Y)-(X+10,Y+10),15*RND,BF
NEXT I
LOCATE 20,1
PRINT "Finish Time = " TIME$
PRINT "Start Time  = " START$
END
```

Debugging. QB3's new debugger is one of its most outstanding features. The capabilities include the ability to step through a program one instruction at a time, set breakpoints, run at a slow speed (called animate mode) watching each instruction execute, and to watch a variable's value as the program executes. It also has a built-in help screen, and the ability to scroll through the source file while debugging.

Turbo BASIC on the other hand has a more conventional form of debugging. It will allow the TRON and TROFF statements, and has a window that can be displayed while the program is running. I found this to be of limited use, however, because the program writes the labels too quickly, and the window will only display labels. Since labels are not required, you sometimes cannot see the desired execution. Overall TB debugging must rely on traditional meth-

ods, such as pauses and printing variables at strategic points within the program.

Defaults. Both QB3 and TB have defaults they use when compiling. QB3 can be set for debugging, setting arrays in row order (as opposed to column order), telling the system if error trapping or event trapping is going to be required, telling the system to minimize string data or not, and setting the compiler to optimize for speed or size. The defaults are debug on, optimize for speed, and minimize string data. A problem with these defaults is that error trapping and event trapping have to be explicitly set if you are using those features. If you forget, the compiler will just flag an error. The thing I don't understand is if QB3 can detect it, why can't QB3 automatically set it?

TB can be set to require an 8087, keyboard break, array bounds checking, overflow checking, stack testing, and trace. The defaults are all off. One thing I would like to see is the ability to change the defaults on a permanent basis. The keyboard checking default is especially annoying, because if you forget to set it, you can't stop your program with CTRL-Break or a CTRL-C.

Assembly Language Interface

A tremendous improvement in both QB3 and TB is the ability to easily interface Assembly Language subroutines into BASIC programs. This ability was sorely lacking in Interpreted BASIC, although a kludgy method of reading machine language code into an array and calling it was available.

QuickBASIC's method of interfacing Assembly Language routines consists of taking an appropriately assembled module and incorporating its object code into a user library with a separate program called BUILDLIB. QB3 programs can then use the Assembly Language subroutine exactly like a BASIC subroutine. The source code for calling DOS or BIOS system services is included with QB3 and the documentation for parameter passing and Assembly Language use is very good.

Turbo BASIC's Assembly Language interface is even easier than QB3's. DOS and BIOS services can be called directly with a

```
CALL INTERRUPT n
```

statement. Registers are set with a REG statement. This is very straight forward and easy. Binary code, generated directly from an assembler can also be easily INCLUDEd into a TB program. Another method of utilizing Assembly Language routines into Turbo BASIC is with an INLINE statement that puts machine code directly into your program where you need it. The documentation for Assembly Language interfacing is excellent.

Listing 5
Mandelbrot Generator

```
DEFINT I-N
SCREEN 9                                    ' 640 x 350 x 16 color graphics
X1 = -2.0                                   ' Upper left corner
Y1 =  1.25
DX =  2.5                                   ' Numerical size of plot
DY =  2.5
DENSITY% = 15

' For interactive input of coordinates, uncomment the following two lines
' INPUT "Input upper left corner : ", X1, Y1
' INPUT "Input width & height    : ", DX, DY

DX = DX/640.                                ' Scale for the screen
DY = DY/350.
START$ = TIME$

FOR I = 0 TO 639
  CR = X1 + DX*I                            ' Real part
  FOR J = 0 TO 349
  CI = Y1 - DY*J                            ' Imaginary part
  A = 0.0
  B = 0.0
    FOR K = 0 TO DENSITY%                   ' Determine count
                                            ' (a+bi)^2 + c =
                                            '     a^2 - b^2 + 2abi + c

      TEMP = A*A - B*B + CR                 ' Calculate real part
      B = 2.0*A*B + CI                      ' Calculate imaginary part
      A = TEMP
      X$ = INKEY$
      IF X$ <> "" THEN GOTO FINI            ' See if operator wants to stop
      IF SQR(A*A+B*B) >= 2.0 THEN EXIT FOR  ' Calculate the size of a+bi
    NEXT K
    PSET (I,J),K                            ' Draw the dot
  NEXT J
NEXT I

FINI:
LOCATE 24,1:PRINT "Start Time = "  START$;  ' Print start and end times
LOCATE 25,1:PRINT "End Time   = "  TIME$;

X$ = INPUT$(1)                              ' Wait for a character to terminate

END
```

Commands

New Commands. One of the most exciting things about the new BASIC compilers is the new commands that are available. Both programs now include full support for all EGA modes with enhanced SCREEN and COLOR commands. Both allow communication with device drivers via the IOCTL command. They also return the lower and upper bounds of an array with the LBOUND and UBOUND commands.

Unique to QB3 are the new commands LOCK and UNLOCK for use in network environments, PCOPY to copy one screen page to another, and SADD to return the address of a dynamically allocated string.

The new commands in TB include BIN$ to return the binary ASCII equivalent of a number, INCR and DECR to increment and decrement a variable, and EXP2, EXP10, LOG2 and LOG10 for base 2 and base 10 computations. It also has LCASE$ and UCASE$ to convert upper or lower case strings, and INSTAT to see if a character is ready at the keyboard without actually reading it. One especially nice feature of Turbo BASIC is the ability to open a file in Binary mode. This allows you to treat any file as a numbered sequence of bytes, without regard to ASCII characters, record length, or other particular file handling details. The current location within the command is set by the new SEEK command. Reads and writes are done with the normal GET and PUT commands. Two other convenience commands available are DELAY, which will delay execution for a set number of seconds without relying upon CPU dependent loops, and MTIMER that uses

your computer's hardware timer to measure time intervals in microseconds up to 55 milliseconds.

Compatibility. Even though most commands look the same in both TB and QB3, they don't always work the same. Most of these differences have to do with screen colors. For instance, in the following code segment:

```
SCREEN 9    ' Select 640 x 350 16
              color EGA mode
COLOR  C
```

Turbo BASIC uses MOD 16 of C for the color, but QuickBASIC returns an error if C is greater than 15. For graphics commands like LINE or PSET, TB uses MOD 16 of the color, QB3 uses color 15. These differences can be easily worked out, but they are differences.

Conclusions

In the final analysis, who is the victor of our BASIC War? The answer is: You are! Both Turbo BASIC and QuickBASIC offer a new, easy to use, professional programming package. The choice for you is between one excellent program and another. If speed and compatibility with other languages is your greatest concern, then QuickBASIC is your best choice. If programming features, a customizable editor, and a superb programmer interface are more important to you, then Turbo BASIC is for you. If you can't make up your mind, you might buy them both. The list price for each is $99, but both can commonly be found advertised for $59 or less.

	TB wo/87	QB	TB w/87	QB87
Compile 579 lines	7.3	7.3	4.5	8.6

Table 3
Relative Compilation Speeds - Seconds

One last comment. BASIC Wars are not over. The final shot is yet to be heard. Both Microsoft and Borland still have a few ideas yet to be implemented, and we will see even better versions of BASIC in the near future. *

The Mandelbrot Set

Bruce Dubbs
122 Valencia Drive
Universal City, TX 78148

The Mandelbrot set is one of those relatively new mathematical functions called fractals. These functions create a strange combination of chaos and regularity from innocent looking algorithms.

The set here is named for Dr. Benoit B. Mandelbrot of the IBM Thomas J. Watson Research Center in New York and was developed in 1980. The first popular description of the Mandelbrot set was in the August 1985 issue of Scientific American.

The algorithm that describes the mandelbrot set is quite simple. To calculate a Mandelbrot value of a point p, a location on the complex mathematical plane is selected. That is, select a point that has a real part and an imaginary part. The imaginary part is a factor multiplied by i (the square root of -1). The following algorithm is then applied:

```
    z = 0
    value = 0
do while ( size of z < 2.0 )
            z = z^2 + p
  value = value + 1
end do
```

The size of z is merely the length of the vector represented by z, or the square root of the sum of the squares of the real and imaginary parts.

For many points on the complex plane, the Mandelbrot value is zero. For some points around the origin, the value is infinite. The interesting areas are in the areas of transition. It turns out that if every Mandelbrot value is assigned a color, a plot of the values returns diagrams that at first appear to be regular, but on closer inspection are not. As an area is magnified, the picture becomes increasingly complex. In fact, successive enlargements of the same region will continue to show new and unique diagrams until we run out of computational accuracy on our computer, or run out of patience waiting for the calculations.

The program shown in Listing 5 shows one implementation of the Mandelbrot algorithm. Photo 1 corresponds to Listing 5. In order to create custom pictures of the Mandelbrot set, uncomment the lines asking for a starting point and window size. The constant DENSITY must also be changed for high magnification in order to allow differentiation of values in your area of interest. Photo 2 shows a start point of -1.0,.35, a window of .1 by .1, and a density of 48.

The program in Listing 5 is written in BASIC specifically for an EGA implementation. The listing is ripe for many modifications. The picture generated could be saved to disk or the Mandelbrot values could be saved to disk. The colors could be modified via PALETTE changes, either dynamically or through user input. Further, a more general program that adjusts to different hardware environments would give many users a chance to explore the wonderful, mysterious world of the Mandelbrot set. *

ENABLE A Start

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

About The Author

George Elwood *is a recently retired Air Force Lt Col. He has been a HUGGER since 1980, when he built his first of two H-89 computers. He has been active in local Heath Users Groups, first in Hawaii and then in Dayton, Ohio. He was instrumental in forming the MS-DOS side of the Dayton Heath Users Group (DAYHUG) when the Z-100s were made available to members of the DoD. He was elected President of DAYHUG and served in this capacity for two years during which the club grew to over 200 members. He is currently employed as the PC director for Logistic Systems Architects Dayton Center where he is developing ENABLE training for both government and commercial markets.*

A few month ago, I wrote a review of ENABLE which was well received. Because ENABLE has so much power, and because of requests, I thought an in-depth look at this package was needed. This is the first of several articles that will deal with ENABLE, each covering only one module of the program with the goal to develop a good understanding of the modules and ENABLE. I will touch upon the integration capabilities as I go through the series with one unit covering this topic in detail. I will also develop a complex report using ENABLE's dot commands to show its advanced capabilities.

ENABLE will probably be the last major software written for the Z-100. It was ported over from the PC version by The Software Group to meet a need and a market that exists in the Department of Defense (DoD with their thousands of Z-100s. Having ENABLE on the Z-100, Z-248 and now on the Z-183 contract will make training easier for DoD organizations. The Z-100 version starts with revision 2.0 and, depending on the support of the Z-100 community, may well have more capabilities added in the future. Because of the development of the Z-100 version of ENABLE, The Software Group received HUG's vendor of the year award at the Sixth HUGCON.

ENABLE has been selected as the integrated package on the Department of Defense's Z-248 and is bundled on the Z-183 contract. It provides word processing, database management, a spreadsheet, graphics, and telecommunications all in one package. Because ENABLE was on the Z-248 contract, some members of the Dayton Heath/Zenith Users Group (DAYHUG) expressed an interest in a group buy. Then when it became known that a Z-100 version also existed and a good price could be gotten for a large buy, DAYHUG members stepped up to it and bought some 130 plus copies. We did, in fact, do some of the beta testing of the Z-100 version, 1.1Z. The Software Group has since upgraded all of the Z-100 versions to the production version 2.0Z.

ENABLE for the Z-100 is a big package. It comes on six disks with about six pounds of documentation. The PC version of the program comes with nine disks of which two are the Perspective 3-D graphics program. You can use ENABLE with a two disk system, but after continued disk changes, you will become very frustrated. It is highly recommended that a hard disk or some high capacity floppy disk system be used. As I said in my ENABLE review, I use a dual 1.2 Meg floppy system for ENABLE, with part of the program on both disks and the data to another disk. The ENABLE setup permits this type of arrangement unlike most large programs.

ENABLE is a very good integration. Unlike some of the other packages on the market, it was designed from the start to be integrated. It did not start as a spreadsheet or database which was retrofitted with the other modules around the base core. The Software Group developed each module as a unit with thought to keeping keystrokes the same in all modules. Although each module was designed by a separate team, they all worked together on the basic rules.

When you receive your copy of ENABLE, you will be impressed by the weight and bulk of the package. It comes with five big manuals (System Overview, Word Processing including spell check, Spreadsheet/Graphics, Database Management, and Telecommunications), two small booklets, a plastic help overlay and a package of six disks (nine with the PC version). One of the short booklets enclosed covers the Z-100 and the difference between PC and the Z-100 versions, and the other is a Quick Reference Guide which is color coded by section. The plastic overlay is two-sided and provides a quick reference for expert commands in all modules. It is convenient and I keep it above the keyboard on my all-in-one Z-120. The initial release of Version 2 has several pages of changes that need to be added to the basic documentation. These pages detail capabilities that were added after the "final" documentation had been sent to the printer. The Software Group has included a device to assist you in inserting these pages. After inserting

these pages you are ready to install the program.

Taking the red folder with the program disks, look for the "INSTALL" disk and insert it in a floppy drive and type "install." The installation procedure for Version 2 has been greatly improved over Version 1.1 or 1.15 on the DoD contract. By following the instructions on the screen, you will go through the entire procedure with displays indicating the next disk to be placed in the drive. There is no need to copy *.*, each disk is in Version 1.1. This is very helpful when it comes to Perspective on the PC version, as it requires its own directory and subdirectory.

The first step is to set up the disk system you are using, hard or floppy. Under the floppy system, you can select 5-1/4, eight inch or the 3-1/2 inch disks. You select the basic directory for ENABLE along with the data subdirectory. The default data subdirectory is "endata" or you can select another name. If you have several users on the system, you can have several subdirectories, with one for each user. This must be accomplished from outside ENABLE, but it is an easy procedure using DOS. It requires creation of these subdirectories from the DOS prompt and creation of .BAT files with the names of the users. An example of the .BAT file would be as follows:

```
cd en20z
enablez (,,,,e:\username)
cd \
type menu
```

The menu would contain the basic data for the selection, i.e., each person would have a number or letter that is the name of the .BAT file. With this structure developed, any user can select ENABLE with one keystroke and have his/her own subdirectory for his/her own files.

After establishing the ENABLE directory, you are prompted for printer type. If you have a hard disk, you can select several types and install the drivers. If you are using a floppy based system, the documentation warns you not to select more than three drivers due to a space limitation on the disk. ENABLE's complete printer driver set is stored as a compressed binary file. When you select a driver, it is written out as a .COM file on the utility disk for floppy based systems or to the hard disk. ENABLE Version 2.0, unlike Version 1.1, permits you to select different printers from the printer menu in the top line menu in the modules. You still have to physically switch between devices, but the printer routines will be available without having to leave ENABLE and re-enter with a different PROFILE.

```
 1 Epson FX80 or FX100                  Q Epson FX80+, FX85, FX286
 2 IBM Graphics Printer                 P ATT 455, 457, or 458
 3 Diablo 630                           Q IBM Quietwriter
 4 Epson MX80 or MX100                  R Toshiba P 351
 5 NEC Spinwriter 3550                  S Xerox 2700 Laser Printer
 6 TI Omni 855                          T Toshiba P 1340
 7 NEC Spinwriter 7710                  U Okidata 192 or 193
 8 NEC PC-8023A-C                       V Diablo C150 or C200
 9 HP Think Jet                         W Fujitsu DL2600
 A Okidata 84 Step 2                    X IBM Color Ink Jet
 B Okidata 92 or 93                     Y HP Laser Jet (Y Font)
 C Prism 80 or 132                      AA Epson JX80
 D Anadex Silent Scribe                 AB IBM Color Printer
 E Okidata 83 with OKIGRAPH             AC IBM ProPrinter
 F Epson RX80 or RX100                  AD IBM Quietwriter II
 G HP Laser Jet                         AE Brother Twinriter 5
 H Qume Sprint or LetterPro             AF Diablo 630 (ECS)
 I Toshiba P 1351                       AG Epson EX800 or EX1000
 J Epson LQ800, LQ1000, LQ1500          AH Toshiba P 351C
 K Printek 900 Series                   AI Diconix 150
 L Centronics PS 240                    AJ Citizen 120-D
 M Epson LX80 or RX80+                  AK C. Itoh M1550S
 N Okidata Pacemark 2410                AL TI OmniLaser 2015
. Select the code of a print driver to be installed or enter if done...
```

Figure 1

ENABLE supports a wide range of printers. Over 50 printers are currently listed. See Figure 1 for the list. Many of the new printers emulate other printers so the actual number of printers supported is higher. An example of this is the ALPS ALQ224, which emulates the Epson LQ1500 or JX80. You would select both drivers during the install process. The Software Group may develop a generic driver builder which will permit the users to build a driver for unsupported printers by asking a series of questions. Because the printer driver routine is stored in compressed binary, it is not possible to modify available drivers.

Once you have selected your printer(s), you are then prompted to continue or terminate the install procedure. If you have the PC version, you are questioned about the video driver you have installed in your system, i.e., CGA, EGA, etc. You can quit the installation procedure or continue. This short procedure permits you to install new printer or video drivers without having to go through the entire installation. The remainder of the installation is straightforward, as you only have to install the disks in the order called for on the screen. The only other option you have is the procedure used to install the tutorial. You can install it on the hard disk, on floppy, or not at all. Perspective graphics on the PC version requires EGA and the install procedure asks if you wish to install it. If you answer yes, a separate directory and subdirectories are created and files copied into the correct location.

Now you are almost ready to use ENABLE on the Z-100. You should check the booklet that comes with the program that spells out the differences in the keyboard on the Z-100. The Z-100 keyboard has many of the features of a typewriter, more so than the PC keyboard. Many of the keys are labeled with a function and The Software Group used them during the conversion. The main difference, and the most important key to remember, is the F0 key. This is the remapped ALT key from the PC version. The key combination SHIFT/F0 results in the PC CTRL key. DO NOT USE THE Z-100 CTRL key for ENABLE's CTRL key. The standard PC keyboard does not have separate arrow keys and number pad. You must use NUM LOCK or SHIFT to get to these functions. This is an inconvenience that I have found when switching between the Z-100 and Z-248. As a long time user of WordStar, I found it hard to change from the DELETE key to the D CHR key. The Z-100 DELETE key will delete from the cursor to the beginning of the line.

Luckily, ENABLE has an UNDELETE function which returns the work that had been removed. Below is a listing of the basic differences between the keyboards.

PC Key	Z-100 Key
CTRL	SHIFT/F0
ALT	F0
DEL	D CHR
CTRL/BACKSPACE	DELETE
INS	I CHR
CTRL/RT ARROW	SHIFT/RT ARROW
CTRL/LT ARROW	SHIFT/LT ARROW
CTRL/HOME	SHIFT/HOME
PGUP	SHIFT/9 or F11
PGDN	SHIFT/3 or F12
END	SHIFT/1
HOME	HOME or SHIFT/7
CTRL/ENTER	LINE FEED
SHIFT/TAB	SHIFT/HELP
SHIFT/PGUP	SHIFT/F11
SHIFT/PGDN	SHIFT/F12

Note that the numbers listed above for the Z-100 keystrokes are located on the key pad and not the numbers above the letters.

You are now ready to start into ENABLE. Because you are a new user, we will go into the tutorials for a brief discussion and then into the program. We will assume that you elected to install the tutorial on the hard disk with ENABLE for this discussion. At the DOS prompt, type "tutor h" (tutor d for disk) and <RETURN>. The ENABLE tutor will now load. Unlike other tutorials, ENABLE's tutor uses the menu and macro capabilities of the program. You will be using a live ENABLE and not another computer aided instruction program, so you MUST follow instructions as they are presented or you will find yourself in ENABLE and wondering how you got there. Using the arrow keys, highlight the area you wish to work in and follow with a <RETURN>. Each lesson takes about 30 minutes to complete and provides an overview of the capabilities of the program.

To enter ENABLE, type "ENABLEZ (,,,,e:\endata),vc". The commas can be used to indicate where certain parts of ENABLE reside and they are as follows: (system disk drive:\path, operation disk drive:\path, tutorial disk drive:\path, utility disk drive:\path, data disk drive:\path). On my system with two high capacity disk drives, I have most of the files on drive C with the utility disk on D. The "vc" indicates a color capability. After getting used to ENABLE, you may wish to add a macro that will bypass the PROFILE selection screen. The macro must be named @0, and it follows the "vc" in the initialization string. You can look at the macro in the tutor as an example of what it is suppose to look like. If you wish to use DOS windows from within ENABLE, you must indicate how much memory will be used by ENABLE with the remaining amount available for DOS operations. If you have ZPC installed, ENABLE will abort with the message that the PC version must be used. You can patch ENABLE so that the check it makes will be bypassed, but the software engineer at TSG that did the Z-100 port says the system will crash if you use DOS windows. ENABLE does not reset the screen after DOS windows, so you will see interesting things.

The first thing you will see is the Sign-On Screen. From this screen you can set the date or time if they are not correct. It is highly recommended that you set the time and date every time you boot up. This gives you a date/time stamp on your files which can be used for back up purposes. If you dislike this procedure, installation of the Smart Clock is fast, easy and inexpensive. After the time, ENABLE asks you if you use PROFILES. If you hit a <RETURN> for this question, you will use the default settings for all operations. The first time through you should use the "F1" key and check/change the default setting on the PROFILE to reflect your normal uses. The F1 key will result in another menu selection where you can CREATE/change or USE profiles. To select the option, you can type the first letter or highlight the option by using the arrow keys. ENABLE handles all menus in this fashion. If you do not wish to work with the PROFILE, you can type SHIFT/1 for the Z-100 or END for the PC. Do not do this before the border appears or ENABLE will abort to the screen prompt.

If you pressed F1 at the sign-on screen, the next screen is the ENABLE Profile Summary Screen. On this screen will be displayed two profiles, DEFAULT and COLOR. You can develop multiple profiles and select them from the Sign-On Screen, if necessary. The up/down arrow combination can be used to highlight one of the profiles listed and will take you to the profile definition menu. Using the arrow keys again, you select the option you wish to revise and hit the <RETURN>. By this time, you can see that ENABLE is very menu intensive. It is possible to use ENABLE without the documentation by using the menu structure alone.

As you move through the hardware section of the PROFILE menu, you will notice a capability listed for serial devices, printers and plotters. On the last page of the Z-100 supplement, it states that ENABLEZ does not support serial devices. A member of DAYHUG discovered that if you map the Z-100 printer/plotter to the serial port under DOS CONFIGUR and select the output device as parallel within ENABLE, it will work. I did this as a demonstration using the Z-100 and the DMP-29 plotter and it did work, in color to boot. It did plot

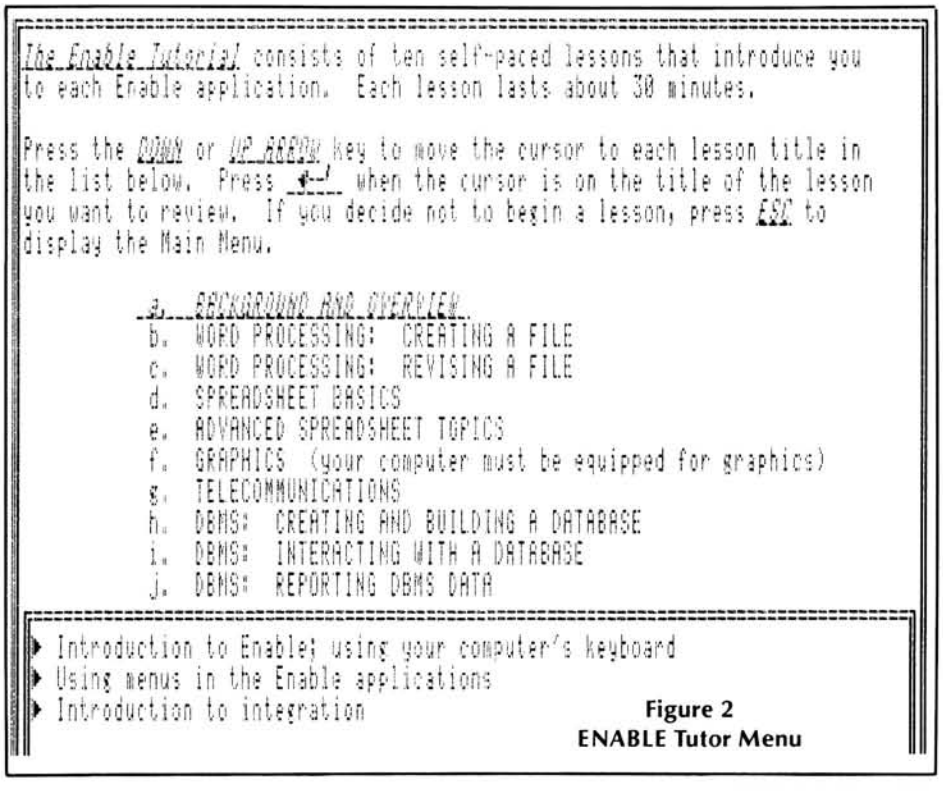


Figure 2
ENABLE Tutor Menu

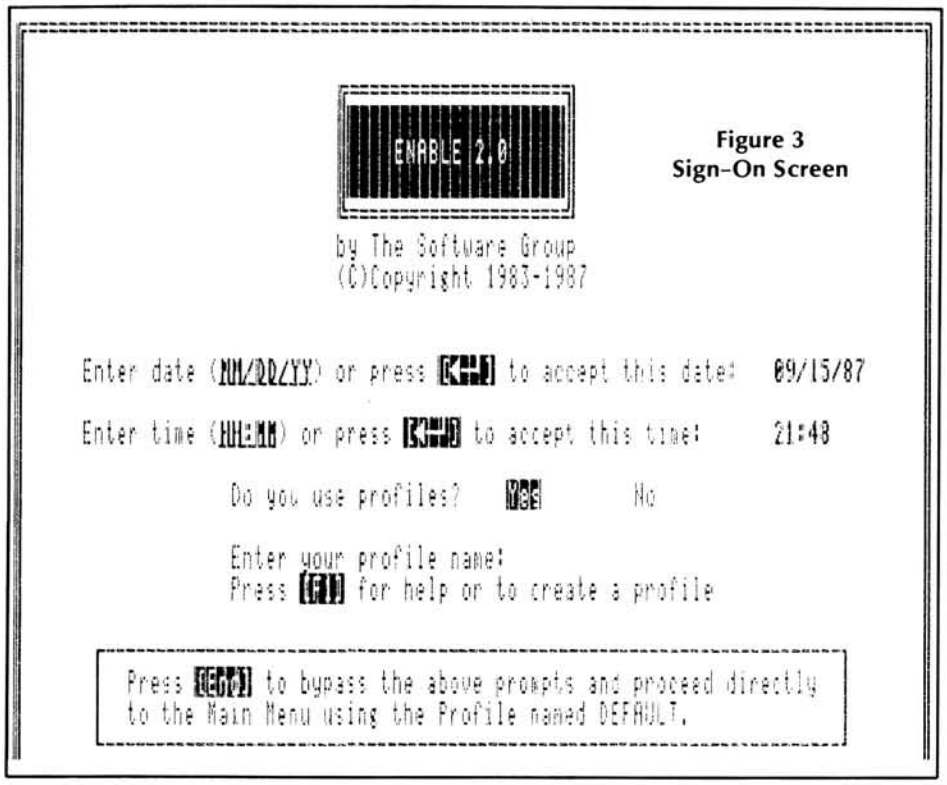


Figure 3
Sign-On Screen

in the large size and because of time constraints, was unable to get the small size to plot. I think that if you select landscape mode for the output that it would work. A serial printer will work the same way as the plotter. This does require extra steps and makes it difficult to work if you have to move back and forth between serial and parallel devices.

The profile definition screen permits you to select many options including the default printer, page spacing, tab settings, color choices to name a few. The Special Text Menu shown in Figure 7 displays the colors that can be displayed during normal operation of ENABLE. Once you have established your default options, these are automatically selected every time you enter ENABLE. Some of these options can be changed, i.e., printer output options. To exit the Profile Summary Screen, hit the F10 key and SAVE the options. You then hit the F10 key again and exit to the Profile menu screen. You can then USE a Profile and continue within ENABLE.

Now you are at the Main Menu of ENABLE. Before ending this session with ENABLE, a little about the screens and the names of the parts that you will see throughout ENABLE. The Main Menu has four options, Use System, HELP, MCM, and Return to DOS. Again, you can select your option by highlighting it or by typing the first letter of the choice. As you move across the line, you will note that the next level of choices is displayed on the line below. Use System will open the four modules of ENABLE for your use. These are the modules to be covered in the next several articles. Help will permit you to use the tutorials. MCM, or Master Control Module, contains ENABLE's system wide features.

The MCM features include windowing, file management, macros, profiles, screen modes including graphics, MCM-DOS interface, and tools. The windowing capability provides you with a means to move between windows, change sizes of windows, and open or close windows. File management allows you to check, delete, rename, copy, check date, time, and size of files. Word Processing files can be displayed in summary showing the creating date, last revision date and total words. You can look by type (word processing, spreadsheet, database management) or all files, again through a menu option. Macros provide the capability to create, revise or list current macros. Another future article will cover macros and custom menu creation. Profiles permit you to do the same functions as previously discussed under opening menu and profiles. Screen modes provides a means to change the mode of the computer between color, B&W, and graphics. The DOS interface provides a means to run programs while in ENABLE and not have to reload every time. From the Z-100 supplement, DOS default is native on the Z-100. You must type "exit" to return to ENABLE from a DOS window instead of hitting the F10 key as for the PC. Tools provide the means to develop the menus and other advanced capabilities of ENABLE. These functions will be discussed in greater detail in a later article. Return to DOS exits ENABLE. If you have a window open, ENABLE will ask if you wish to exit or save the work in progress.

The Top Line Menu is the first line you see

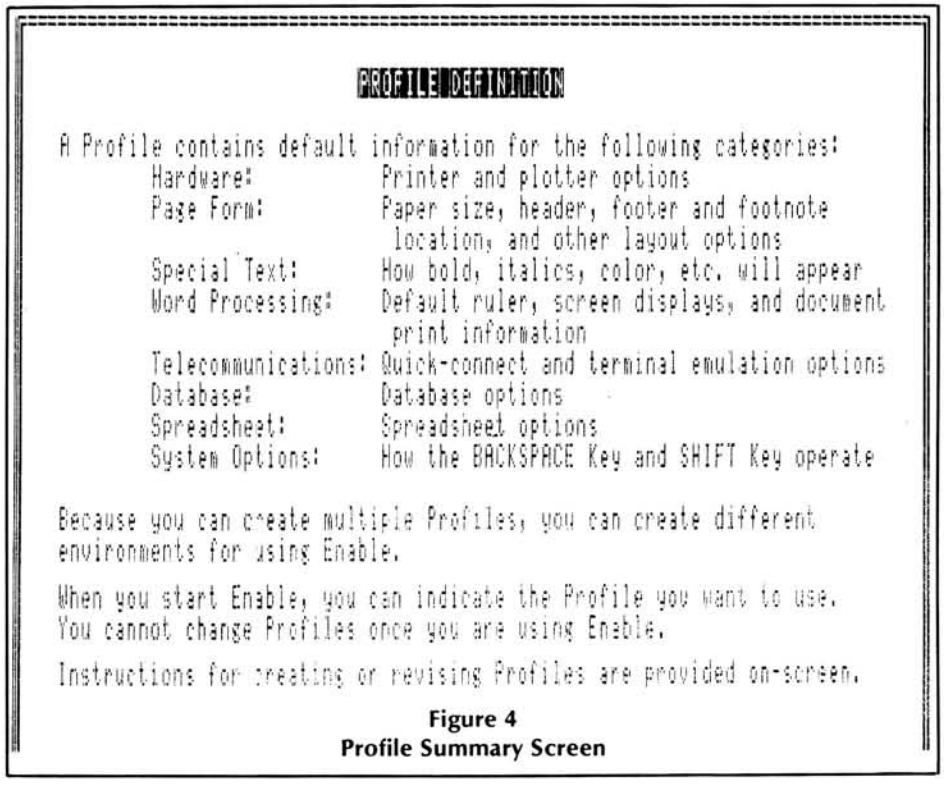


Figure 4
Profile Summary Screen

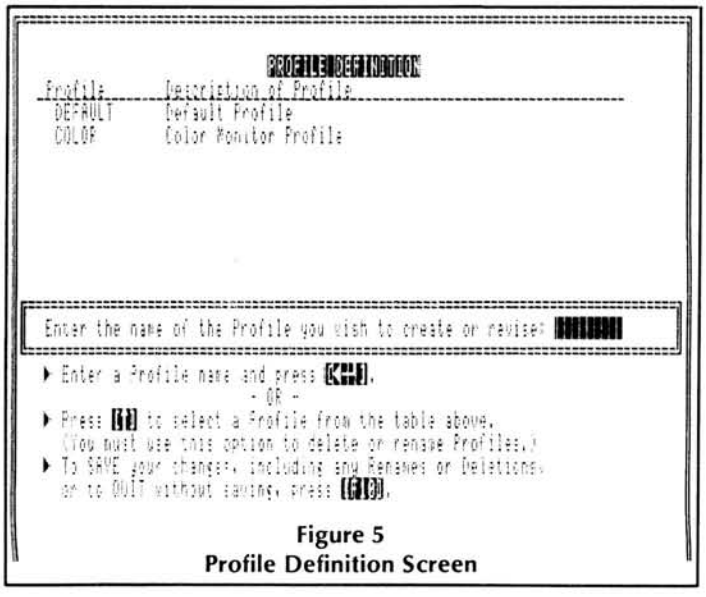


Figure 5
Profile Definition Screen

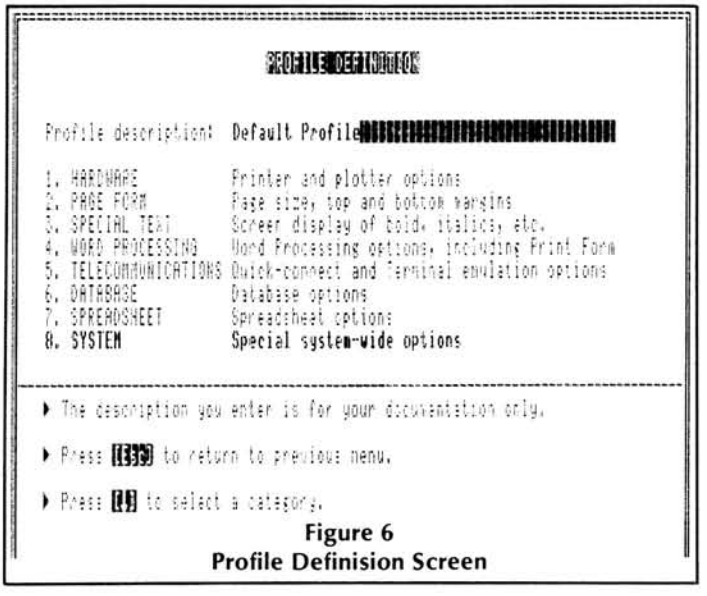


Figure 6
Profile Definision Screen

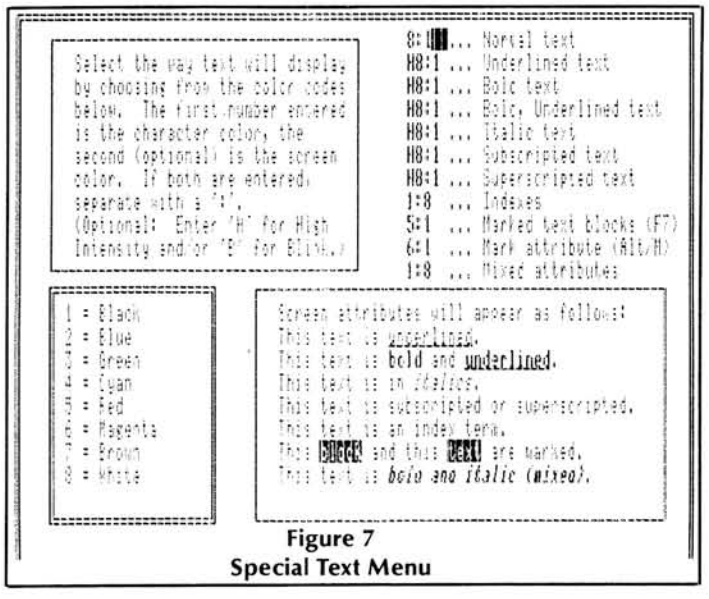


Figure 7
Special Text Menu

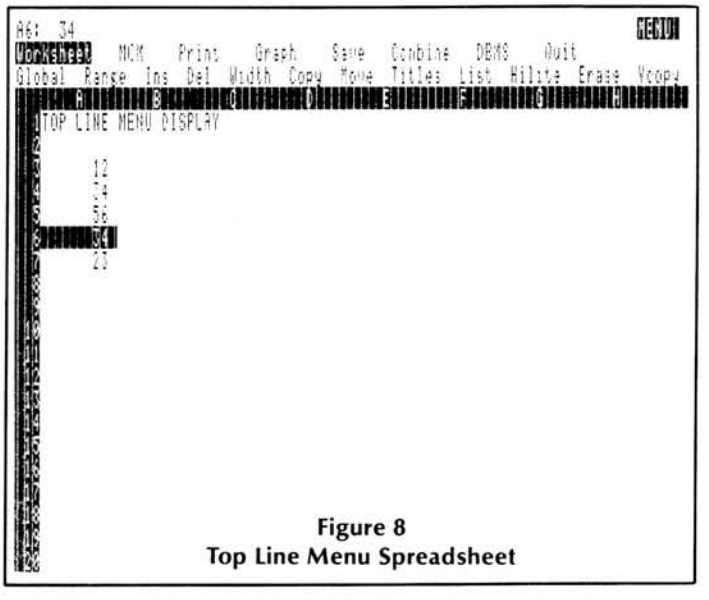


Figure 8
Top Line Menu Spreadsheet

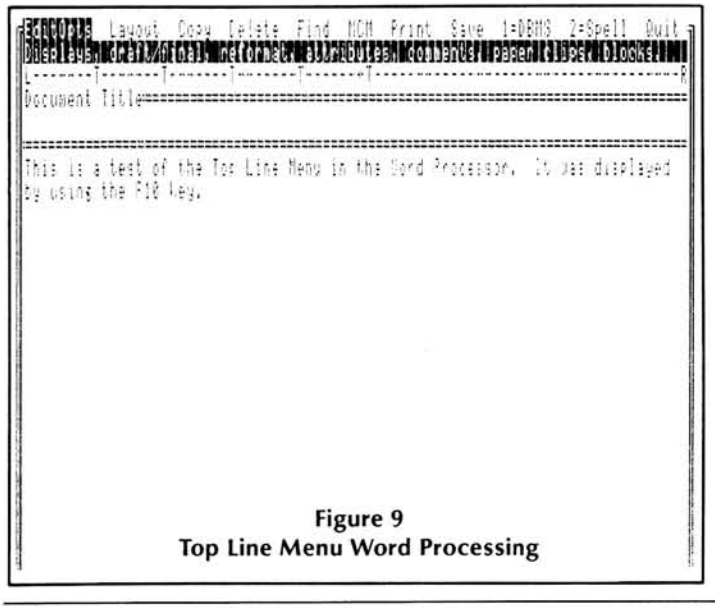


Figure 9
Top Line Menu Word Processing

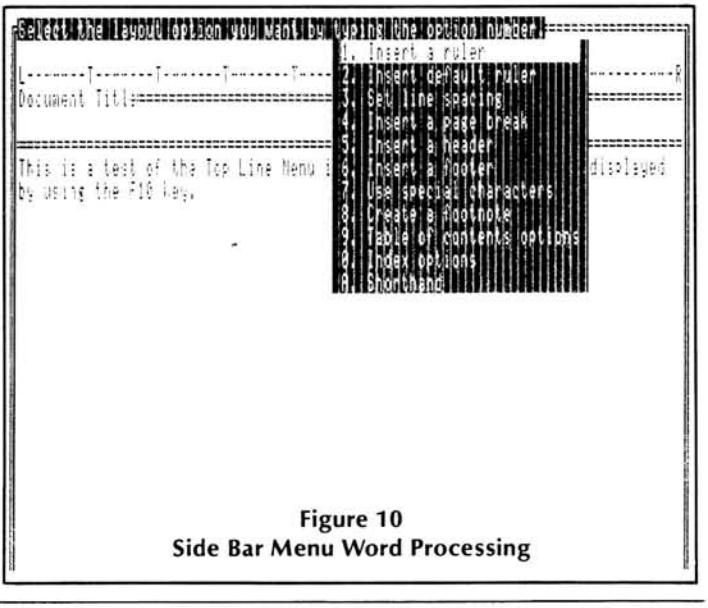


Figure 10
Side Bar Menu Word Processing

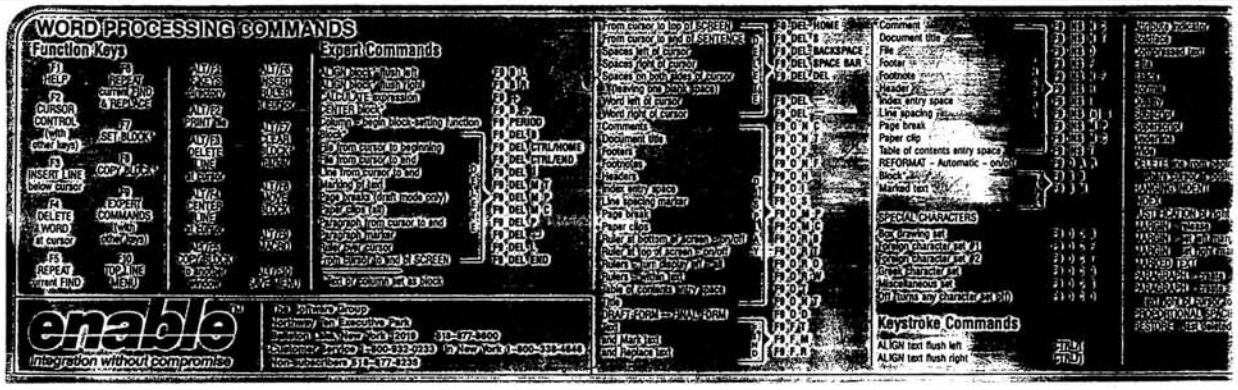


Figure 11
Part Of The Overlay

in the various modules of ENABLE. They can be displayed by hitting the F10 key at any point.

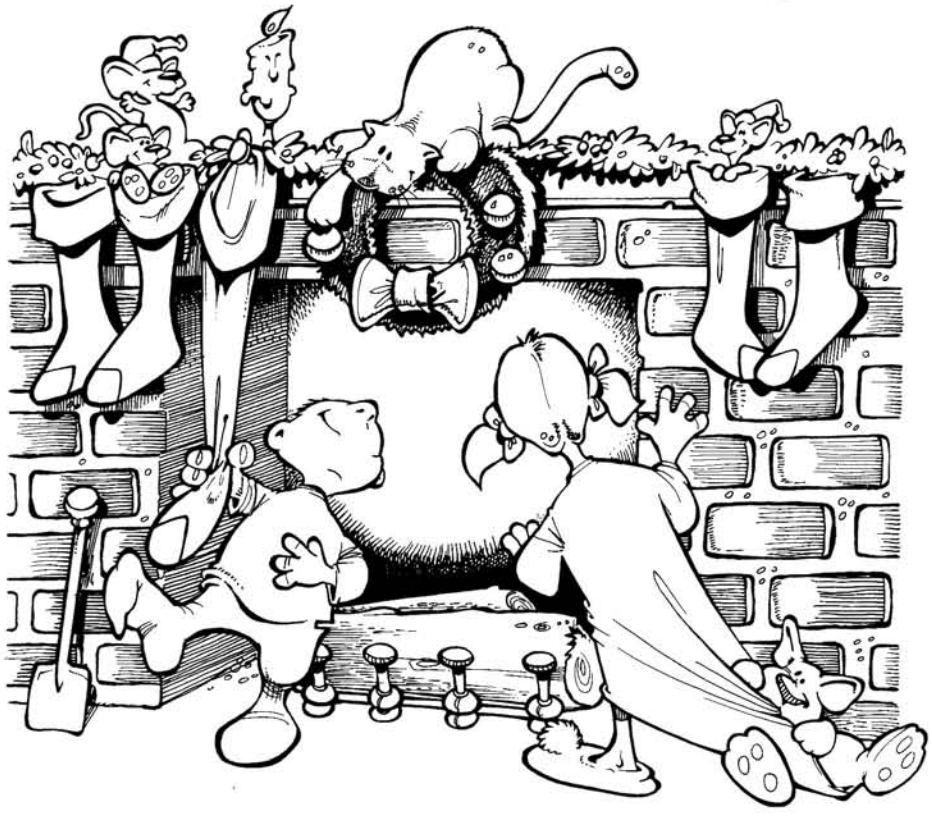
The Side Bar Menus appear under the Top Line menu. When you select the Top Line Menu, the Side Bar Menu is displayed showing the options under each choice.

As you become proficient in ENABLE, the menu structure can be by-passed by using the expert commands as mapped in the function keys. There functions are listed in the Quick Reference Guide and on the plastic help overlay.

The last line is the Status Line. This line contains the status of ENABLE in the mode you are operating. It shows the window number, file name and disk drive location, display mode, any attributes in use during word processing, and the cursor location (line and character). There is a problem in the PC Version 2.0 with the Status Line. The line is displayed in bright white letters on a white background while in the Co80 mode with an EGA card. This is hard to read, but by going to the graphics mode, the letters are black on white. There is not a problem with the Z-100 version.

You have now installed ENABLE and are at the opening screen and are ready to do some productive work. The next article will cover word processing and spell checking. ✱

On The Leading Edge



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

Christmas Gifts, IBM's PS/2 Computers And OS/2, Zenith's OS/2, Hard Disks, Bump And Grind

As they (whoever "they" are) say, "time flies when you're having fun", and I've had so much fun this year I can hardly stand it. Many projects, including two new FlipFast books, have kept me busy and out of trouble (well, busy anyway). There have been a number of interesting events that have occurred this year, and I will mention a few that I think are significant for us. And I will continue the idea of suggesting some low-cost ideas for Christmas gifts for computer users that I started last year. Most of these have been chronicled in these pages in the last 12 months, so I will not go into significant detail on any one of them — all are recommended; some are highly recommended. Lest I forget, let me wish all of you a "Merry Christmas" or "Happy Holidays", whichever you prefer. There is always a question about what kind of gifts are useful to computer users and hobbyists. If you want to drop a subtle hint to your Santa Claus, you can do so by circling the appropriate item in this article. All ordering information, including prices, for specific items mentioned here are listed at the end of the article. Some of these items are on my "Gotta Have" list for various reasons, and they will probably be on yours, too. For the most part, I have limited these ideas to $100 or less. Now let's start with some general ideas.

General Supplies

For one reason or another, computer systems seem to eat floppy disks. That doesn't mean they are destroyed by a computer, but there always seems to be a need to buy more. Floppy disks come in all sorts of shapes and sizes, and they make good stocking stuffers. Depending on the size and format, you can generally get good floppies for under $20 for a box of 10.

Printer supplies are another item. Paper is an obvious idea, and you will probably want to get the paper that has the fine perforations (e.g. Micro-perf or equivalent) because it is easy to remove the edges. A box of paper generally runs under $40 depending on where you buy it.

Printer ribbons are another consumable that can generally be purchased for under $10 each. There are so many types of ribbons that the only caution is to buy a good brand.

If you have a daisy wheel printer, new daisy wheels are always useful. My DTC Style Writer printer that is similar to the Brother HR-15 uses the Brother daisy wheel cartridges that I can usually find for under $25. Although the daisy wheels for my printer are plastic, you might find some that are made of metal for your printer. Their extended life is probably worth the additional cost. Because I use my daisy wheel printer a lot, I have already had to replace several cartridges because one or more of

the "daisys" (letters) broke off after extensive use.

HUG Software

One of the advantages of being a HUG member is the excellent software library that we have access to. This high-quality software is either developed by HUG staff or contributed by Huggies in one way or another. Two particular representatives of this software library come to mind.

First, there is Pat Swayne's HEPCAT — the HUG Engineer's and Programmer's CAlculation Tool, for long. This is a dynamite memory-resident pop-up calculator that is compatible with virtually all software. It provides standard floating point calculations for sine, cosine, tangent (and their "arc" values), natural and base 10 logarithms (ln and log, plus "anti" functions), not to mention the conversion from English to metric units. Of course, HEPCAT supports scientific notation that you would expect in an engineering calculator. For programmers, calculations can be done in a number of radixes, including both hex and binary, not to mention decimal. HEPCAT even does logical operations in binary, such as AND and OR. I have used this in place of my hand-held calculator because it is more convenient to use and has more power for calculations. In particular, HEPCAT is much easier to use for calculations in non-decimal radixes. I like HEPCAT, and I think you will too.

If you are an advanced DOS user, you will appreciate the features and capabilities of HADES — the HUG Absolute Disk Editing System. This program is NOT intended for the novice user, although many beginners may find it useful to explore the mysteries of the DOS disk formats. Current DOS disk formats are documented in my latest MS-DOS FlipFast book, in case you are wondering where to find out the peculiarities of each disk format. HADES does precisely what its name implies; you can see and edit anything on a disk, including the directory and the File Allocation Tables (FATs). With HADES, you can change just about anything on a disk, but the downside of that is that you can also completely destroy a disk. In short, you must know what you are doing, but I have found it to be an excellent program. You can also use HADES to recover an erased file from a disk. If you are interested in exploring or fixing disks, this is the program for you.

General Software

It is impossible for me to resist a plug for the new Zenith/Heath FlipFast books, since I had some input to both of them. The first one is for the Zenith/Heath GW-BASIC which was written by Bill Barden. My contribution to that book was the technical editing for the specific BASIC version offered by Zenith. Aside from the detailed descriptions of the BASIC commands, statements, and operators, Bill has included a number of excellent ideas on streamlining BASIC code so that your programs will run faster and more efficiently. As usual, many of these tips and tricks are not included in other documentation anywhere. This 408 page book is jam-packed with all kinds of useful information for BASIC programmers.

My new book on Zenith/Heath MS-DOS includes all commands and other information up to version 3.20. It is a 544 page book that contains a compilation of just about everything I know about DOS. It was specifically written to be useful to a beginning DOS user, as well as the most advanced DOS user. If you are a beginner, this book can help you graduate to the ranks of the real "power users". I have been told that ALL users particularly like the fact that ALL error messages are documented in this book, and a cross reference listing is provided in case you are not sure what program generated which message. In many cases, I have found that error messages included in my book are not documented anywhere else, including the Zenith manuals. Where possible, I have also included tips on recovering from errors.

One other feature that I have received a lot of compliments on is that I tried to show USEFUL examples of how each command can be really used. I find that most books talk a lot about options, but they simply do not show or describe how a command is commonly used. I tried to demonstrate and explain the how and why of the more common uses of each and every command.

Several programmers have told me that they really like all of the programming information that is included. Over 100 pages of the book are specifically devoted to technical information that is difficult to find, not to mention the usual programming commands like EXE2BIN, MASM, and LINK. Have you ever tried to find a listing of scan codes for the PC keyboard? Both PC and AT keyboard scan codes are included, and I also added the Z-100 Escape Sequences for us die-hard Z-100 fans. And if you do any programming in BASIC (or any other language), you will find that this programming information can be invaluable.

Both books include information for both the Z-100, as well as the PC compatibles. They can be ordered direct from the publisher using the address listed at the end of this article. I am sure that all of you recognize that both of these books are clearly in the "gotta have" category that is highly recommended for everyone. Like many authors, I admit to a kind of "cheerful self-appreciation" that allows to recommend my work because I believe that it is the best. That is a clearly unbiased opinion since my friends know that I am quite objective in my evaluations. If you will pardon me for a minute, I think I will need to remove my tongue which has somehow become firmly embedded in my cheek. I can't imagine how that possibly happened!

Barry Watzman's Perks programs are another excellent choice for utility software. The original Perks runs on the Z-100, and Perks PC is for the Zenith PCs and other Zenith compatibles, including IBM and Compaq. Perks is a "desktop accessory" program that provides features like a calculator, notepad with cut and paste to applications, card file, telephone dialer and XMODEM communications, and a calendar for appointments among others. Perks has all kinds of neat features (not all were mentioned) that make it an appropriate and thoughtful gift for just about any Z-100 or PC compatible computer user.

Another favorite of mine is WindowDOS, and I have noticed that Joe Katz likes it, as well. WindowDOS is essentially a file manager that has lots of additional features. Although it can be run like any program, it was designed to be memory-resident so that you can use its features any time. Once loaded, it is activated by CTRL-INS that displays a directory. The cursor keys are used to move around in the alphabetically sorted file list that also includes subdirectories. As each entry is highlighted by the cursor key movement, the top of the display shows the file's creation date, time, and attributes. You can change subdirectories by simply moving the cursor key to a name and pressing RETURN. This is an invaluable feature for moving through a number of subdirectories on a hard disk so that you can delete unwanted files.

WindowDOS also includes commands that can be executed by a single letter such as COPY, DIR, ERASE, FORMAT (floppies only), GLOBAL, LIST, MKDIR, RENAME, SORT, TREE, and VIEW. These commands have some really neat features. For example, ERASE will not only erase a file, it will also "erase" a subdirectory name when it is

empty like DOS should have done. All 10 Function Keys are used for various purposes, including an on-line help function. If you have a hard disk, you can even use WindowDOS to back up files with no problem because it can copy files to multiple disks. You can easily "tag" multiple files for any function (like copy or erase) by pressing the plus (+) key on the keypad. All in all, a real neat program that is useful to just about any computer user. Unfortunately, WindowDOS is only available in a PC version.

If You Have A Hard Disk

There are a wide variety of hard disk utilities available, but my first choice is the MACE Utilities. If you have a hard disk, it may have a warranty, but it is absolutely guaranteed that, at some point, it WILL fail. Hard disks are complex mechanical devices, and when they fail, there is a catastrophic loss of data because of their data storage capacity. That notwithstanding, there are all kinds of "user errors", such as the old DEL *.* trick that can wipe out megabytes of data in a few seconds.

The MACE Utilities provides a way to "undelete" files, and that works on both hard and floppy disks. Even if you have a floppy disk system, you will probably find that you will need to recover an erased file at some point. Also included is the REMEDY program that allows you to check for and lock out bad sectors on a disk. I have used REMEDY extensively on my Z-200 system, because I had the misfortune to buy several boxes of "auto-crash" BASF disks that contain anywhere from 60,000 to 100,000 bytes of bad sectors.

If you have a hard disk, it is necessary to reorganize the directory entries and FAT chains to optimize hard disk performance. I run these utilities about once a week. All of these features (and others) make the MACE Utilities indispensable, and this software should certainly be a top priority for all hard disk users. As you might expect, MACE Utilities is only available for PC compatibles.

There are many ways to backup a hard disk, but my favorite is certainly the DSBACKUP+ program that I use for both my Z-100 (ANSI version) and my Z-200. DSBACKUP+ is fast, reasonably priced, not copy protected, and easy to use. I think there ought to be a law against copy protected programs that are used to backup a hard disk — I don't care how good the program is. In any case, I particularly like the fact that DSBACKUP+ can be run from a menu or direct from the DOS command line. By the way, the ANSI version requires that the Z-100 ANSI device driver (e.g., ANSICON .DVD or equivalent) be installed in the CONFIG.SYS file. The DSBACKUP+ program is much better and faster than the DOS BACKUP, and I also include it on my "gotta have" list.

Hardware

There is one inexpensive hardware item that is particularly appropriate for most computer users, and that is a hardware clock. These "real-time" (or no-slot) clocks keep track of the current date and time, so you do not have to reenter them every time you boot your computer. They plug into the ROM slot in your computer and are entirely self-contained including a battery. And they work just fine in a Z-100 or PC compatible computer. Of course, you don't need one of these if your computer already has a built-in clock, like the Z-200 or IBM AT.

I use the Zenith Smart Watch in my Z-100, and I really like it. One caution about that however; the Zenith Smart Watch includes three spacers and longer screws for the video board in the Z-100, because there is not quite enough clearance for the Smart Watch. If you have a Z-100, I particularly recommend the Zenith Smart Watch because I know these extra parts are included. Other similar units may also have these, but I have not checked them. In any case, once you have one of these little jewels, you will wonder how you got along without it.

For Programmers

I admit that my favorite programming is C. If you are looking for a more powerful language than BASIC, I recommend C because it is several orders of magnitude more functional. If you like to do "bit-fiddling", C is particularly recommended. C is usually considered to be a high-level language, and as such, it is much easier to learn than assembler. C also gives you the capability to access to low-level hardware features like assembler does. Perhaps the most important feature is the fact that C compilers are available for all kinds of computers, including mainframes, and that means the code is portable.

Assembler is hardware unique which means that you need to learn a new assembly language every time a new family of microprocessor chips is introduced. For example, the 8-bit assembly language used for the Z-80 (in the H-89) is considerably different than the 8086/8088 assembly language used in the PC compatibles. And even though the 8086/8088 assembly language is generally used in today's PC compatibles, new microprocessors (like the 80386 CPU) are continually being announced that have additional hardware features. I expect that good C compilers will provide access to these features, and I have a suggestion for that.

My favorite C compiler is the Ecosoft C-88 compiler. In addition, Ecosoft provides a wide variety of support for that compiler that is useful both for the beginner and the advanced C programmer. Ecosoft's president, Jack Purdum, has written four books on C, and they cover everything from beginning to advanced C programming. In addition to the C compiler, you can get a debugger, windows library, librarian, and even a Computer Assisted Instruction course on C from Ecosoft. All Ecosoft products are highly recommended and are listed at the end of this article.

A final note about all of these ideas. I have checked all of these products and have found them to be quite useful. In other words, these are not products that I have looked at and retired to a shelf — I actually use them in my work.

PS/2 In Perspective

It is not possible to wrap up this year without making a few comments about IBM's announcement of the PS/2 computers and the OS/2 operating system. From all reports, the PS/2 systems have not received a particularly enthusiastic response, especially the Model 80 system which is IBM's contribution to the 386 computer line. Since all of the other systems are essentially reworks of existing technology, there is not a whole lot to say about it. The Model 80 is different in several ways.

First, the Model 80 has an 80386 CPU that distinguishes it from the rest. That is an obvious hardware difference that does not have any particular compatibility impacts, except for speed of processing. The second major difference is something called the Micro-Channel bus that is used to connect the various components in the system, such as add-on boards. Current reports are that this bus is really a turkey.

One reason is that the Micro-Channel bus is NOT compatible with any of the existing add-on boards. That is not too surprising because the 80386 CPU needs to have a 32-bit bus for optimum performance. Existing computers, like the Z-200 and the IBM AT, have used the old 16-bit bus originally designed for the IBM PC. But there is something else.

The connections for the boards on the Micro-Channel are MUCH closer together than they are on the old bus. As I recall, there is about 0.10" separation between the board connections on the old bus. On the Micro-Channel bus, the tolerances are measured in thousandths of an inch. I have seen reports in InfoWorld that several hardware vendors have "smoked" a few motherboards on these new systems which has been attributed to these close tolerances. I suspect the fact that IBM's documentation had a "minor" error about which voltage was available on what pin (also reported in InfoWorld) may have had something to do with that, as well.

IBM has reportedly developed the Micro-Channel bus to help reduce the competition from other computer manufacturers. I doubt that will work because there are all kinds of ways to accomplish the same end, and I think it is a little late in the game to be introducing a new "standard".

There have also been reports that some of the existing software does not run on the Model 80. I exclude the copy-protected software, since that is known to have problems on a lot of computers. The bottom line is that this new "standard" does not seem to be PC compatible, so I would urge caution before getting too involved with the Model 80, in case you are inclined to do so.

OS/2 In Perspective

OS/2 is a clever acronym for Operating System/2. Despite reports to the contrary, it is highly probable that most compatible manufacturers will make OS/2 available for systems that are able to handle it. That means, you will need a computer that has at least an 80286 CPU just in order to run OS/2. In addition, I strongly suspect that a hard disk will be nearly mandatory because OS/2 is rumored to be quite large in terms of disk space requirements — about three megabytes or so.

In addition to IBM, both Zenith and Compaq have announced (in InfoWorld) that they will provide OS/2 for their systems. Perhaps most interesting is that the Zenith announcement said that they would be shipping OS/2 by the end of this month. Other reports indicate that Microsoft does not have "final" code (as of the end of September) for OS/2, so that may have an impact on Zenith's announced shipping date.

One other report that seems to be consistent is that OS/2 will be expensive, probably a minimum of $300 or so. And I have also seen reports that the "Extended" version may cost in the neighborhood of $900 or so. Seems like an awfully expensive operating system for a microcomputer to me. I would expect to see the documentation in gold-plated three-ring binders at those prices. Nobody knows where this will stop, so I guess we will just have to wait and see. But if the Zenith marketing people were smart, they would price their OS/2 so that it was less than IBM's.

Although there have been other reports about OS/2, I will not comment on any of them until I have seen it. I believe that these other reports that I have mentioned are probably true, but I would not want to make any specific comments until I have actually seen how OS/2 runs. In any case, stay tuned for more information on OS/2. I expect to be writing about that in at least two articles next year.

More On Hard Disks

There are a lot of ways to implement a hard disk in a system. One of the popular ways seems to be using one of the cards that contains a hard disk and the hard disk controller. Gary Brindley (Panama City, FL) wrote to ask me which one to get.

There is no easy choice about these "hard disk cards". I think that they are more expensive than the technology warrants, and you will pay a premium price for it. In general, you will find that the hard disk cards are more expensive than buying a separate hard disk and controller. Perhaps the only reason for buying a hard disk card is just the ease of installation. Some people say that you cannot have two floppy drives and a hard disk in a '151 computer. Not true. You can buy a half-height hard disk that fits just fine, but you will not be able to see the red "access" light. I don't consider that a disadvantage because the hard disk cards obviously do not even have one. But you can save a considerable amount of money if you are willing to pick up a screwdriver and disassemble your computer.

My suggestion for buying any hard disk is to buy it from a reputable source. After looking through the September REMark, I find a number of advertisements that you might find helpful. Payload Computer Services (page 6), Scottie Systems (page 44), Graymatter Application Software (page 46), and RAM Technology (page 80) all advertise hard disks. I also know that long-time Heath/Zenith supporter First Capitol Computer (page 60) sells hard disks although they were not advertised in this particular issue.

Fred Griffee (Annandale, VA) reported that he had all kinds of interesting "problems" implementing a hard disk on his '151 with a Western Digital and an Omti controller. There are several points to be made about this. After the hard disk and controller are installed, use the Zenith PREP program to prepare the hard disk. Both the Omti and Western Digital controllers provide a low-level formatting program that can be used, but PREP is much better because it tests the hard disk. Fred also found out that you canNOT change controllers without rerunning their low-level format programs. One other point is that the '151 dip switch that sets the autoboot feature for the hard disk does not work with the Western Digital and Omti controllers. That should be no surprise because the XT computers do not have that feature, and the controller ROM's do not check it for that reason.

The Bump And Grind

I have received several letters about the computers that seem to do the Bump and Grind. Hard disks do the Bump and Grind — some of them do it quite well, in fact. WHY they do it depends on several things. For example, the 26 megabyte Tulin hard disk in my Z-100 does the Bump and Grind when the system is powered-on. After getting the Tulin manual, I learned that this was because the hard disk performs self-diagnostics when the power is turned on. These diagnostics take about 10 seconds to run, but the hard disk makes a number of unusual noises during the diagnostic testing. So, if your hard disk makes unusual noises when the system is powered-on, it may be because it is performing self-diagnostics.

The Bump and Grind also seems to depend on the capacity and manufacturer of the hard disk. The 20 megabyte hard disk in my '200 makes all kinds of interesting "ker-thunk" noises during read and write operations. On the other hand, the 40 megabyte drive is so quiet that I cannot hear it during a disk access. In many cases, the Bump and Grind is normal for a hard disk and does not necessarily mean that there is any kind of problem like an impending failure. If the hard disk made the same noises when it was new, you can reasonably assume that it is normal for that hard disk.

There is one other very common reason that a hard disk system can perform the Bump and Grind. If you use a controller that is not manufactured by Zenith, your system will almost invariably do one of the best Bump and Grind routines that I have ever heard. The light for drive A will come on, and the system will grind for a few

seconds before it boots from the hard disk. You can thank the IBM "standard" for that. The first time I booted an IBM XT computer, I have to admit that I was quite worried about all of these strange things. Here is what the system is doing when you have a controller that is not manufactured by Zenith (e.g., Omti or Western Digital).

Most important is the fact that non-Zenith controllers totally ignore the DIP switch setting for the autoboot feature on the '151 and other similar computers. Since the IBM computers never had that feature in the first place, they had to use another technique. For some reason, these controllers are so "dumb" that they cannot detect the presence of a hard disk in the system. So, the system ALWAYS attempts to boot from drive A. No discussion, no exceptions, and no fooling. If it cannot boot from drive A, then it tries to boot from a hard disk. Hopefully, that will be successful. If you have a non-bootable disk in drive A, and try to reboot one of these systems, you will very cleverly be told that you have a "non-system disk" in the drive. Then you have to open the door on drive A and reboot the system.

I have never been able to understand why other hard disk controllers have not been able to get around this problem, but I suppose there is some obscure reason for it. In any case, the Z-200 SETUP program eliminates the problem because you have to specify the boot sequence for the computer.

In The Mail

I occasionally receive a letter from a Huggie who is surprised that I use a handwritten letter to answer a question or whatever. One recent reader letter mentioned that he was "surprised that someone like me would use a handwritten reply" to his letter. There are several reasons for that.

My assumption is that since you have taken the time to write, you would like a reply as soon as possible. I try to answer all letters that include a self-addressed stamped envelope so that you receive the answer within a maximum of three weeks after you mailed it — usually within two weeks depending on the mail. Since I have a pretty busy schedule, that means I have to take advantage of time whenever it is available. Frequently, that means I am not at home at my computer. For example, I am teaching three classes in a row on Tuesdays and Thursdays this semester at UTA beginning at 8:00 AM. I usually have a few minutes "dead time" between classes which would normally be wasted, but sometimes I answer letters while I am waiting for the next class. As a matter of fact, I usually take some correspondence with me whenever I leave the house so that I have something to do with time that would otherwise be unused. Sometimes I make notes about the answer to the letter; more often I simply write the letter to you so that you will get a faster reply. I have even made up my own "correspondence pack" that contains everything I need to answer letters, and I carry it with me most of the time when I leave the house. Even though I admit that my handwriting is not the best, I have been told that it is "communicable", and I can usually read it, too.

I learned about this trick in a "time management" class that I took a number of years ago. My "correspondence pack" includes a note pad, and I sometimes make article outlines or notes for books in addition to writing letters. For those of you interested in the idea, ALL notes for "planned correspondence" (letters that I initiate) and unanswered letters go in this "pack" so that I know where they are. The "pack" is nothing more than an 8-1/2" × 11" notepad clipboard with a pocket in the fold-over cover. It also contains stationery, envelopes, and usually a few stamps, just in case.

I sometimes carry my REMark folder so that I can make notes for upcoming articles. This folder is one of those soft cardboard two-pocket deals that are frequently used by students. I usually have to replace it about once a year because it gets quite a bit of use as you can imagine. It contains a copy of all unpublished REMark articles so that I can remember where I am. And it also contains notes for future articles that I may have made during some other "free" time (between classes). Many of the subjects that I write about are a direct result of questions in your letters, since I assume there is a general interest in that topic.

Other ideas for articles come from the students in my classes and are based on questions they ask about computers in general. Since a good part of the classes that I teach focuses on the use of PC-type computers, this helps me keep in touch with the concerns of beginning computer users. And since UTA uses the IBM PC with PC-DOS, I can keep up to date on the latest IBM DOS changes and bugs.

Powering Down

There still seems to be a considerable interest in hard disks, so I will continue to write about them. Next time, I will talk about disk access time and its impact on your system. This was part of the presentation that I gave at the HUGCON in August. In the meantime, I still appreciate your letting me know about your questions and interests — keep those cards and letters coming. As I mentioned in this article, I know something about other computer systems, and that is the reason that I like the Heath/Zenith combination the best.

If you have any questions about anything here, be sure to include a self-addressed, stamped envelope (business size preferred) if you would like a personal reply to your question, suggestion or comment. And I wish a Happy Holiday season for you and your family.

Products Discussed

Hardware

Smart Watch (CB-5063-50) $59.00
Heath Company Parts Department
Hilltop Road
St. Joseph, MI 49085
(800) 253-7057 (Heath Catalog orders only)

Software

DSBACKUP+ $79.95
Design Software
1275 W. Roosevelt Road
Chicago, IL 60185
(800) 231-3088 (Orders only)

Eco-C88 C Compiler (MS-DOS)	$ 59.95
Flexi-Graph (Graphics Support)	39.95
Librarian	29.95
C Library Source ($10.00 if ordered w/C-88)	20.00
Developer's Library (All source) ($25.00 if ordered w/C-88)	50.00
ISAM Library ($15.00 if ordered w/C-88)	30.00
CED C Program Editor (PC only)	29.95
C Self-Study Guide (Purdum) (+ $2.00 shipping)	17.00
C Programming Guide (Purdum) (+ $2.00 shipping)	20.00
C Programmer's Library (Purdum) (+ $2.00 shipping)	22.00

Ecosoft, Inc.
6413 N. College Avenue
Indianapolis, IN 46220
Orders only: (800) 952-0472
Technical Support: (317) 255-6476

FlipFast Guide to Z/H GW-BASIC	$21.95
FlipFast Guide to	24.95

Z/H MS-DOS
S-A Design Books
515 W. Lambert, Bldg. E
Brea, CA 92621-3991
(714) 529-7999

HUG Software
HEPCAT (885-3045-37) $35.00
HADES (885-3040-37) 40.00
Heath/Zenith Computer Centers
Heath/Zenith Users' Group
Attn: Nancy Strunk
P.O. Box 217
Benton Harbor, MI 49022-0217
(616) 982-3838 (HUG Software only)

MACE Utilities version 4.10 $99.00
Paul Mace Software, Inc.
400 Williamson Way
Ashland, OR 97520
(800) 523-0258 (Orders only)

Perks
Z-100 only $69.95
PC Series and IBM 69.95
Barry Watzman
560 Sunset Road
Benton Harbor, MI 49022
(616) 925-3136
Master Card and VISA accepted

WindowDOS $49.95
WindowDOS Associates
Box 300488
Arlington, TX 76010
(817) 467-4103 *

Building An 8 MHz H-151

Update

Robert Maskasky
3220 Carlton Court
Greene Cove Springs, FL 32043

Since the publication of my article in the May issue of REMark, I have received several letters and phone calls from people who have had troubles with running their H/Z-151 or H/Z-161 computers at 8 MHz. The problems seem to be of three basic types.

The first is that many of the Zenith floppy controller boards will not accept programming from the CPU at 8 MHz. This problem causes the floppy disk drives to lock-up, operate unreliably, or not to boot-up from floppy disk. This problem is easily cured by adding another I/O wait state. Wait states are a method used by Intel to enable the CPU (8088) to use inherently slow memory or Input/Output devices. Everytime the CPU accesses the memory, the CPU issues the memory address onto the memory bus and then checks the "READY" signal to see if the memory has the data ready. If the "READY" signal indicates that the memory is not ready, the CPU will wait until the memory is READY. The CPU waits in multiples of the clock pulses. Each additional clock pulse caused by a NOT-READY condition is called a wait state. Obviously, each wait state slows the speed of the processor. In most personal computers, the "READY" signal is generated by a circuit which assumes the memory is operating at a certain speed and generates the necessary signal to automatically add a predetermined number of wait states. The H-151 adds no wait states to the memory operations. The Input/Output operations of the CPU also use the "READY" signal to slow the speed of the CPU for slow I/O devices. The H-151 assumes that the I/O devices require one wait state. Input or Output instructions are only about 5% to 15% of the instructions executed by the processor. Therefore, adding I/O wait states have an almost undetectable effect on the processor speed.

In the Z-151, the "READY" signal is buffered and synchronized to the clock by the Clock Generator part number 8224 (U211 in the Z-151). The READY input to the 8224 is "RDY88" and "DMA-ACTIVE". The "DMA-ACTIVE" signal is associated with the Direct Memory Access and is not directly associated with the problem. The "RDY88" signal is generated by the wait state generator circuit (U226 and U227) which generates the necessary signal to cause the processor to wait for one clock cycle on each I/O operation. The memory operations still will operate with no wait states, so that you can realize the full benefit of the 8 MHz speed.

The key to adding I/O wait states is to rewire the wait state generator circuit. Fortunately, this can be accomplished simply by removing U227 from its socket and adding a couple of jumper wires to the chip prior to re-installation. This can be accomplished by adding wires directly to the chip, or by using a header plug (Radio Shack part number 276-1980) and soldering the chip to the header plug.

Before beginning the modification, I suggest you purchase a spare 74S174 chip in case you damage the chip during the modification. The 74S174 costs less that $5.00 at most electronic supply stores. Then, carefully cut off pins 10 and 13 about 1/8 inch from the body of the chip. Using a low wattage soldering iron, solder a piece of 22 gauge wire between pins 10 and 13. Next, cut another piece of 22 gauge wire about 1-1/2 inches long. Solder one end of the wire to pin 12 of the chip (close to the body of the chip). Do not allow the solder or the wire to damage the part of the lead which inserts into the socket. Now insert the chip back into the U227 socket. Insert the loose wire end (connected to pin 12) into the socket U227 pin 10. Do not allow the wire to short against the chip pin 10 or any other pins. Check the chip pins 10 and 12 to ensure they are not shorted to the socket pins 10 or 12. Also inspect the wires for shorts. I suggest you try the computer at 4.77 MHz to ensure the computer is still operational. Next, try the computer at faster speeds and ensure the floppy disk controller operates properly. You are now operating at 2 I/O wait states. If you are still having troubles with the computer, try adding another wait state. This can be done by disconnecting the wire from U227 pin 12

(connected to the socket pin 10) and connect it to U227 pin 15, so that U227 pin 15 connects to the socket pin 10.

The second problem area is with certain chips not being capable of the 8 MHz speed. I am referring to the use of 120-nanosecond memory chips and an 8 MHz 8088-2 processor or an 8 MHz V-20 processor. Also, the 8237 and the 8255 chips must be replaced with the 8 MHz Harris 82C37 and 82C55 chips. I have also found that the AMD D8288 is the best chip to use for U243. I had previously thought the Harris 82C88 was best, but I have found the AMD part to be slightly faster. Usually, the extra speed of the AMD part is not necessary, but, in some marginal machines, the AMD part may help. I suggest you try your machine at 8 MHz with the two I/O wait states, without changing the 8288 chip. If your machine does not work, try an AMD D8288.

The third problem area is with expansion boards. Some of these boards will not operate at 8 MHz. If you are having troubles, I suggest you remove all the expansion cards from your computer and try it again. I have found that some of the inexpensive EGA boards, hard disk controllers, and some memory expansion boards have problems at 8 MHz.

Periodically, when rebooting the computer using the <Ctrl><Alt><Del> key combination a "-- CPU FAILURE --" message will appear. This is caused by the clock changing speeds during the CPU operation. The two clocks on the turbo board (4.77 MHz and 8 MHz) are not synchronized and during the speed switch, a narrow clock pulse may be generated and upset the operation of the CPU. Since this occurs during the CPU TEST portion of the BIOS, the BIOS may detect and report a CPU failure. If this problem occurs, a hardware reset will clear the problem, or turning the power off for a few seconds also works.

Software. I have had some complaints about the "DISKFIX.COM" program which I published in the May '87 REMark. For some unexplained reason, changing the Verify function (Interrupt 13h Function Code 04) to a Read function (Function Code 02) does not work on all Z-151 machines (possibly a hardware speed related problem). The BIOS verify function is rarely used. This function is used by the "FORMAT", "DISKCOPY" and some disk utility programs. This problem does not affect the operation of the "/V" flags on any of the software I have tested. The "/V" option used by these disk utility programs usually involves a data read to verify each bit of the data on the disk with the data that was written to the disk. The /V option usually does not use the troublesome BIOS verify function. I wish that Zenith would take the hint and modify the BIOS to eliminate this problem.

I have modified the DISKFIX.COM for those people who cannot use the original version. The modified version simply acknowledges a request for a Verify with a Verify Good (no error) response. I have provided a new feature to the "DISKFIX.COM" program. The program now locates the floppy disk parameter table, and modifies the parameters for the maximum speed for your particular floppy disk drives. The floppy disk parameter table is pointed to by memory location 0000:0078 hex (interrupt vector 1E hex). The ninth byte in the floppy disk parameter table is the Head Settling Time (in milliseconds). The tenth byte in the floppy disk parameter table is the Motor Start Time (in increments of 250 milliseconds). These numbers are established by the BIOS during boot-up and are set very conservatively to ensure the machine will work with floppy disk drives of various speeds.

The Head Settling Time is the time delay for the floppy disk head to move from track to

track and this delay will occur 40 times during a format of a 360k byte floppy disk. The Motor Start Time is the initial time delay when the floppy disk drive is first accessed (from when the access LED comes on to when the actual read starts) this delay occurs only once per disk access.

By changing these parameters, you can customize the machine speed for the maximum speed which your floppy disks are capable. The best way to test if the speed selected is too fast is to try to format a disk with your machine set to run at the fastest CPU speed. If you get a "Track 0 Bad...Replace disk...." error message, immediately depress the return key fooling the machine into thinking that you have changed the disk. If you do this before the floppy access LED goes out, the Motor Start Time is not necessary since the motor is still running. If the format proceeds without error, increase the Motor Start Time. Otherwise, try increasing the Head Settling Time. The -14 BIOS sets the Head Settling Time to "0D" hex (about 14 milliseconds) and the Motor Start Time to "04" hex (about 1 second). I tried TEAC 55B floppy disk drives and Mitsubishi disk drives with the Head Settling Time at "00" hex and the Motor Start Time at "03" hex. Using these parameters, MS-DOS 3.2 and a 8 MHz CPU, I can format a 360k byte floppy disk in about 35 seconds. Remember, that every time you make a patch to the "DISKFIX.COM" program, you must re-run the program. Since the program has a Terminate and Stay Resident portion, you must reboot the machine to re-run the program. This program should work with any version of MS-DOS or PC-DOS. For more information about these time delays, I suggest you read the "MS-DOS Programmers Utility Pack" and "Faster Disk Access" by Pat Swayne, REMark, August 1986.

The revised "DISKFIX.COM" program is provided in two formats, the Assembly Language version and the data necessary for creating "DISKFIX.COM" using the MS-DOS "DEBUG" program.

Listing

```
PAGE 60,132

;******************************************************
; INTERRUPT 13 TURBO SPEED FIX-UP ROUTINE
; ASSEMBLY LANGUAGE SOURCE LISTING
; BY  Robert J Maskasky
;******************************************************

CODE SEGMENT

ASSUME CS:CODE,DS:CODE,ES:CODE

ORG 100H

START:
      JMP INITIAL                   ;INITALIZE THE INTERRUPT VECTORS

;*****************************************************************
; FLOPPY DISK (INT 13h) HANDLER
;
; Function is to intercept INT 13H, Function 04h and return a verify
; good (no error occurred)
;*****************************************************************

BEGIN:
      CMP AH,04h                    ;TEST FOR VERIFY FUNCTION
      JNE INT_13_CONTINUE           ;IF NOT VERIFY FUNCTION

      CMP DL,04h                    ;TEST FOR WINCHESTER OPERATION
      JA INT_13_CONTINUE            ;IF NOT FLOPPY FUNCTION

      CLC                           ;Set the NO ERROR flag

      DB 0CAh                       ;IRET AND SAVE FLAGS
      DW 02h

INT_13_CONTINUE:
      DB 0EAH                       ;JUMP TO INTERRUPT HANDLER
NEXT_OFFSET:  DW 0
NEXT_SEGMENT: DW 0

;*************************************************************
; End of Terminate and Stay Resident Portion
;*************************************************************
; INTERCEPT THE DISK I/O INTERRUPT VECTOR
;*************************************************************
INITIAL:
      MOV AX,3513h
      INT 21h                       ;GET Floppy INTERRUPT VECTOR
      MOV WORD PTR NEXT_SEGMENT,ES
      MOV WORD PTR NEXT_OFFSET,BX

      MOV DX,OFFSET BEGIN           ;SET THE TRANSFER ADDRESS
      MOV AX,2513h
      INT 21h

;*************************************************************
; SET THE FLOPPY DISK PARAMETERS FOR MAXIMUM SPEED
;*************************************************************

      MOV AX,351Eh                  ;GET THE FLOPPY DRIVE PARAMETER BLOCK
      INT 21h
      MOV DI,BX
      ADD DI,09h                    ;POINT TO Head Settling Time CONSTANT

      MOV AL,00h                    ;MOTOR START TIME=00
      MOV AH,03h                    ;HEAD SETTLING TIME=03 Milliseconds

      STOSW                         ;STORE THE NEW CONSTANTS

      MOV DX,OFFSET INITIAL         ;EXIT AND STAY RESIDENT
      INT 27H

CODE ENDS

END START
```

For those people not familiar with Assembly Language programming, the DISKFIX program can be created using the DEBUG program, as follows. To create the DISKFIX-.COM program, type the following items as written in Italics (x represents don't care response):

```
DEBUG
-NDISKFIX.COM
-E100
xxxx:0100  xx.EB xx.14 xx.90 xx.80 xx.FC xx.04 xx.75  xx.09
xxxx:0108  xx.80 xx.FA xx.04 xx.77 xx.04 xx.F8 xx.CA  xx.02
xxxx:0110  xx.00 xx.EA xx.00 xx.00 xx.00 xx.00 xx.B8  xx.13
xxxx:0118  xx.35 xx.CD xx.21 xx.8C xx.06 xx.14 xx.01  xx.89
xxxx:0120  xx.1E xx.12 xx.01 xx.BA xx.03 xx.01 xx.B8  xx.13
xxxx:0128  xx.25 xx.CD xx.21 xx.B8 xx.1E xx.35 xx.CD  xx.21
xxxx:0130  xx.8B xx.FB xx.83 xx.C7 xx.09 xx.B0 xx.00  xx.B4
xxxx:0138  xx.03 xx.AB xx.BA xx.16 xx.01 xx.CD xx.27  xx.89
xxxx:0140  <Return>
-RCX
cx 0000
:003F
-W
writing 003F bytes
-Q
```

The byte at address 0136 is the Motor Start time and the byte at address 0138 is the Head Settling Time. Change these bytes as necessary to support your floppy disk drives.

For those people who like to experiment, the INT 13h routine usually involves a call to INT 40h which does the actual work with the disk drives. Some hard disk controllers will intercept the interrupt 13 looking for hard disk operations. If the requested disk operation is a floppy disk request, the hard disk controller forwards control to interrupt 40. The DISKFIX.COM program will also work if "INT 40h" is substituted at the "INT 13h". The program patch is in location "011F"h and "012F"h. You can change the "13" to a "40" in these locations and your program will work with INT 40h. In the Assembly Language versions of the program:

Change:
"MOV AX,3513h" to "MOV AX,3540h"
Change:
"MOV AX,2513h" to "MOV AX,2540h"

A Few Words About BIOS. There are numerous versions of the MFM-150 BIOS ROM used in the H/Z-151. The most recent version is identified by a "-14" stamped on the top of the ROM chips (U207 and U208 on the CPU board). Another way to identify the BIOS version is to use the <Ctrl><Alt><Ins> key combination to enter the MFM-150 monitor. The -14 BIOS is monitor version 2.9. Some of the earlier versions had compatibility problems with some software and expansion boards. The main complaint I have heard about is the -11 BIOS and some ABOVE BOARD memory expansion cards. Zenith denies that any problems existed with the -11 BIOS, however, the -14 BIOS seems to have solved these problems. For BIOS updates, I suggest you contact your local Heath/Zenith dealer. The new BIOS chips cost about $25.00 for the two chip set. Sorry, Zenith does not offer the Z-151 BIOS in a single 256K (27256) version, and the Z-158 BIOS does not work well in the Z-151.

I have found that most of the EGA boards will not support the MFM-150 monitor "TEST" function. This function is implemented by using the <Ctrl><Alt><Ins> and typing "TEST". On many EGA boards this results in a blank display. In "Computer Shopper", July 1987, "Expanding your H/Z", Ted Drude reports that on the CPU board, SW1 positions 4 and 5 should both be placed to the right (on) position for EGA compatibility.

Conclusions

I have tested several Z-151 computers at 8 MHz with no memory wait states and a V-20 processor. Norton Utilities "SI" rates my machine at 3.1. I have researched some of the other speed modifications available for the H/Z-151 computers and found that most of them run the computer at about 7.4 MHz. At 7.4 MHz, the H-151 rates at 2.7 on the Norton Utilities "SI". Interestingly enough, the "Wildfire" modification by Software Wizardry rates at 2.7 to 2.8 on the "SI". Software Wizardry did not solve the problem of the "FORMAT" or "CHKDSK" at high speed and does not allow the <Ctrl><Alt><Del> reboot or software selectable speed (ref Sextant, April-May '87). The Z-158 rates about 2.5 on the "SI" when using a V-20 processor and 1.5 when using the 8088 processor. The 8 MHz Z-151 is about 25% faster than the Z-158 and about 10% faster than "Wildfire". When used with a V-20 processor, the 8 MHz Z-151 is running about twice the speed of the IBM-PC, about 30% slower than the 6 MHz PC-AT and about half the speed of the 8 MHz Z-248. I feel that 8 MHz is the practical limit for the Z-151. Faster speeds may be possible, but faster speed crystals are hard to find and at speeds faster than 8 MHz, many expansion boards will not work. At 8 MHz, the Z-151 computer no longer seems to labor with word processors, such as Wordstar. The machine still seems slow with CAD programs, large spreadsheets or databases, but so do all personal computers. If you are still not happy with the speed of the Z-151, I suggest purchasing a Z-248 computer which rates at about 7.7 on the Norton "SI", or wait for the rumored new Zenith computers, which use the 80386 processor and will probably run "SI" of about 16 to 20. *

Multiplan Update

Richard (Rich) L. Mueller, Ph.D.
11890-65th Avenue N.
Maple Grove, MN 55369

Introduction/Overview

What I would like to do in this article is to highlight some of the features that came out in Microsoft Corporation's Multiplan V3.0, and mention a few that were a part of V2.0 since some of you still may be at V1.2. This article will cover Multiplan as it relates to both the H/Z-100 series and the H/Z-100-PC series. In addition to the well-documented features, I want to point out some features which are either buried or undocumented, as well. Multiplan is one of the few Microsoft products that will work on an H/Z-100 machine, as well as PC compatibles as it is released by Microsoft.

Back when I got my H-100 in early 1983, one of the first application packages that I purchased was Multiplan V1.02. In June of 1985, I purchased the update of Multiplan from Zenith Data Systems to V1.2. I believe V1.2 was the last version released before V2.0. In fact, if my memory serves me right, at the time I was able to get the update to V1.2 from Zenith, V2.0 was already released by Microsoft.

In the Fall of 1985, I wanted to get a version of Multiplan for my Z-160 and wanted the latest release, so I went out to a mail-order house and purchased what was called the IBM-PC version of Multiplan V2.0, since Zenith had no idea when they were going to make V2.0 available to their users. When I went to install it on my Z-160, I found out that this particular version actually supported a number of microcomputers, including the H/Z-100. This is also true with V3.0 which I purchased recently.

The installation program allows the user to either select a microcomputer that has already been defined by Microsoft or to define his/her own micro by providing control sequences for function keys, graphics characters, positioning the cursor, clearing the screen, initializing Multiplan, etc. The Multiplan program itself is modified by the install program based on the micro selected and/or information provided. The PC micro and compatibles is the default micro supported. However, the H/Z-100 is one of the many other micros that both Version 2.0 and 3.0 support. So the user does not have to define his/her own terminal; just select the H/Z-100 from the microcomputer menu.

Requirements

My old V1.02 manual states that version only requires 64K of RAM, while V2.0 requires 128K as a minimum. Version 3.0 doubles that and requires 256K. The more memory, the larger the worksheet that can be supported. Remember that Multiplan requires the entire worksheet to be in memory when it is active. Multiplan V3.0 still only requires one disk drive to work. Also, MS-DOS/PC-DOS V2.0 or higher is required.

Files Supported

Before discussing the worksheet enhancements and various new features, let me comment on a few changes that were made to the way files are handled by Multiplan. First of all, Microsoft made some changes in the format of Multiplan binary files (known as Normal Multiplan files) at V2.0 and V3.0 time. According to the Multiplan V2.0 manual, one can convert V1.02 and V1.2 binary files to V2.0 by simply using the TRANSFER LOAD command to load an "old" file. Then use the TRANSFER SAVE command to save the loaded file using another file name. The new file will be in V2.0 binary format.

The above worked just fine on my Z-160 and I was able to convert all my old files to the new format. However, I was not so lucky with my H-100. I kept getting errors when I tried to load an "old" file. At this point, I decided to first convert all my "old" files using V1.2 to the Multiplan Symbolic SYLK format. This format is selected via the TRANSFER OPTIONS command. Then I loaded V2.0 and loaded the SYLK files which worked just fine. Finally, I saved them under V2.0 as V2.0 binary files.

Nowhere in the manual was there ever a mention that a possible problem may exist when trying to read "old" files on V2.0. I called Microsoft about this and the person that I talked to acted as though there was

no such micro as an H/Z-100. It was suggested that I try exactly what I already had done, via the SYLK file route.

The Version 3.0 manual states that the TRANSFER LOAD will read previous Multiplan files and when one performs a TRANSFER SAVE, the file is saved in V3.0 format. Please note that previous versions of Multiplan cannot read V3.0 files. Since I had already converted all my V1.2 files to V2.0, I don't know if the same problem that I had on my H-100 with going from V1.2 files to V2.0 exists with going from V1.2 files to V3.0. If the problem is still there, the SYLK file route is the solution.

In V2.0, there was a special utility called CONVERTD that converted one file type to another. One of those file types was ASCII. In V3.0, this utility is gone. For ASCII files, there is a new subcommand for the TRANSFER command called IMPORT. This will allow the user to load an ASCII file. To save a file in ASCII format, requires the user to PRINT the file to a disk file rather than to the printer. For the other formats supported (DIF — Data Interchange Format — Visicalc Files, SDF — dBASE II Symbolic Data Format, Lotus 1-2-3 and Symphony Files, and SYLK — Microsoft's Symbolic Link Files), Multiplan V3.0 determines automatically the format when loading the file.

Multiplan V3.0 allows the user to save worksheet files in standard Mutilplan binary files, SYLK files, ASCII files, Lotus and Symphony files. The method for saving a file in ASCII format is via the PRINT command as stated above. For SYLK file format, the user selects the Symbolic option in TRANSFER OPTIONS. For the Lotus and Symphony files, the proper file extensions determine the final format (e.g., .WKS for Lotus V1A and WK1 for Lotus V2.0).

Please note that when loading a non-Multiplan spreadsheet, everything in those files is not converted to Multiplan, such as Lotus macros which are not compatible, some functions, etc. However, the main part of the worksheet (spreadsheet), including the data, is converted to a Multiplan file. One can then go in and make whatever changes are necessary to make the worksheet function properly under Multiplan.

Macros

A macro facility was added to Multiplan V2.0 and enhanced in V3.0. It is a major feature which was lacking in earlier versions of Multiplan. Since it is a major enhancement in my opinion, I am going to cover this subject in some detail.

Macros allow the user to define a set of keystrokes (command steps) to be executed by just calling the macro name or ID. Macros in any language or application make the job easier, reducing errors, reducing tedious tasks, eliminating many repetitious tasks, etc. In other words, macros make our work easier and, hopefully, more efficient.

For Multiplan, the user writes macros by entering a string of characters in a cell or consecutive cells in column fashion. Each cell can contain up to 255 characters. Each character or sequence of characters represents a step in executing a Multiplan command. For example: To format a cell, one would press "F" to select the format command, then "C" for cells, followed by a range of cells (or a single cell), etc. The "F", "C", and "range of cells" can be entered as a string of characters in a macro.

Some characters require a special way of specifying them, because they are either special functions, such as "backspace" and "return", or there is no single character to represent them, such as "word left", "word right", etc. These special functions are represented by what Microsoft calls "key codes" which consist of an apostrophe followed by two letters. For example, the "return" is represented by 'rt, the "backspace" by 'bs, etc.

The user has two ways of entering these "key codes". The first way is to explicitly enter the apostrophe followed by the two-letter code. The second way is via the "Edit Macro" key (CTRL-V, or F5 on the PC), which toggles the meaning of the special function keys. When the toggle is on (Multiplan is in EM mode — Edit Macro mode), pressing the "backspace" key will not backspace the cursor, but will enter the appropriate "key code": 'bs. Similarly, with all the other special functions, which number over two dozen. To return to normal mode, press the "Edit Macro" key again to toggle this special mode off.

After entering your entire macro, end the macro with either a blank cell or the special "key code" sequence: 'qu.

Although it is not necessary to name a macro or give it a macro ID, it is, however, strongly recommended that this be done. Naming a macro makes it easier to find at a future date or the name can be used to invoke the macro. Likewise with the ID, a quick way of invoking the macro can be made using this ID.

To invoke a macro via the two-letter ID, press the "Alternate" key followed by the two-letter ID. The "Alternate" key on either the PC compatibles or the H/Z-100 is the CTRL-A. On the PC compatibles, the ALT key can be used in place of the CTRL-A.

A macro can also be invoked via the GOTO command. The GOTO command allows one to specify either the name of the macro, if one was given to the macro when it was created, or to specify the reference (Row/Column position) of the upper left cell of the macro range. The macro can be terminated at any time via the "Cancel" key. This is either the ESC or CTRL-C keys.

To help with the checking out of a newly created macro, Multiplan provides a single-step feature. Press the "Single-Step" key (see the Keyboard section for details) to put Multiplan in this special mode (you will see an ST on the status line). Then start the macro by either of the two methods described above. After the execution of each step of a command within the macro, Multiplan will stop. This will allow the user to see whether the macro is performing what he/she wanted it to do. Pressing any key will start the execution of the next step. To take Multiplan out of single-step mode, press the "Single-Step" key again.

At times, there is a need to execute a series of commands every time a worksheet is loaded for whatever reason. To automate this process, there is a special macro that has the name "AUTOEXEC". Just as with MS-DOS/PC-DOS where the AUTOEXEC.BAT file of commands will be executed each time the system is booted, the AUTOEXEC macro will be executed automatically each time the worksheet containing this special macro is loaded.

Briefly, here are some other features of the Multiplan macro facility. There may be times when it becomes necessary to enter some information from the keyboard while executing a macro, so Multiplan provided an interactive capability. This allows the user to create macros that can perform different tasks based on input from the keyboard.

There is also a subroutine-like capability which allows one macro to "call" another. The second macro, when finished, will return back to the calling macro. In addition to the special "key codes" that were discussed earlier, there are a few special ones that allow the user to put comments in a macro, to put out messages to the display screen, to help with the generation of menus, etc.

Now let me give you a couple of real simple macros for examples. The first macro contains the following character string (explanation follows):

```
IC1'rtFW12'rtFCC'tb'tb$'rt'qu
```

What this macro does is this. At the current column position (wherever the cell pointer is positioned), insert 1 column (IC1'rt), format the width to be 12 characters wide (FW12'rt), and format cells in the entire column to have the $ notation (FCC'tb'tb$ 'rt). The first C in this last character sequence is to select formatting of cells. The second C specifies the entire column rather than some range of cells involving rows and columns. The final special "key code" in the macro is to end it.

The second example is an AUTOEXEC macro. As was stated earlier on the PC type macros, it is possible to change the foreground and background colors of the worksheet. Although, the color selections can be made permanent, I chose to load the worksheet in the default black background and white foreground characters. After the worksheet is loaded, this sample AUTOEXEC macro changes the background color to blue (so we have white characters on a blue background). This macro may not be a very useful macro, but it nevertheless shows how a macro can be automatically called and executed at the time a worksheet is loaded.

```
WP15'tb1'rt'qu
```

This macro (must have the name AUTOEXEC) selects the Window command (W), followed by the Paint subcommand (P), the 15 is the white foreground color, and the 1 following the tab is the dark blue background color. Try it on a PC compatible like your H/Z-100-PC machine.

In Multiplan V3.0, an enhancement was made to the Macro facility: now there is a way to record keystrokes as the user is executing some task. These keystrokes are saved as a macro for later playback. In V2.0, the user had to enter every character into the macro definition; now with V3.0, the user can let Multiplan generate the actual macro by turning on the 'Record Macro' feature.

To utilize this new enhancement, the user simply names a range of cells (column-wise) as a macro just as in V2.0. However, the name must be RECORD. Instead of entering the macro characters, the user turns on the Record Macro feature via the Record Macro key (see Keyboard section for details). Then the user performs whatever operations he/she wants to capture. When done capturing the keystrokes, turn Record Macro off. Change the name of the macro to something other than RECORD and delete the RECORD name. The new macro can now be executed.

Other New Features

Date and Time data is now possible with V3.0. Not only can Date and Time values be entered and recognized as such, but also they can be used in calculations. In order to recognize information as a date and/or time, the cell(s) must be formatted to hold this kind of data. Internally, the date and time is stored as a decimal number (Multiplan refers to this as a serial number). The digits to the left of the decimal point represent the number of days since January 1, 1900. The digits to the right of the decimal point represent the time in seconds since 12:00:00 midnight (the decimal number is actually the total number of seconds since midnight divided by the total number of seconds in a day). For example: serial number 31926.5057523148 equates to 29 May 87 12:08:17.

The user doesn't have to worry about any calculations. Multiplan takes care of that. The user selects the format of the date/time cell from a menu of many and Multiplan displays the information accordingly. The date can be displayed as all numeric, Months spelled out completely or first 3 letters, days of the week can be displayed in full or just the first 3 letters, etc. Time can be displayed in AM/PM format, 2400 format, only hours, only minutes, etc. If one of the predefined formats is not acceptable, the user can define his own format.

An Audit feature has been added in V3.0. Auditing is the process of verifying or tracking errors in formulas, call references, identify named ranges and macros, and examine worksheet statistics to identify errors. Formulas are audited to verify that a result is being calculated correctly, and to identify the cause of an error value or message. The AUDIT CELLS command is used to determine what formula a cell is used in.

Closely related to auditing is the Reports (V3.0 feature) that can be generated. First, there is a cross-reference report that shows how cells refer to each other on the entire worksheet. Only cells that contain formulas are included in this report. A second report allows the user to get a list of all names in a worksheet. This list can contain all macro names, or all named areas, or both groups. The third report possible is one that will list unusual conditions that most likely will lead to errors. Some of these conditions are: formulas that refer to blank cells, cells that have error value messages, cells with formats and no values, etc.

Data Encryption is now available with V3.0. The user can save a file in encrypted format by specifying a password for the file. The password itself is used in some kind of algorithm that is used to encrypt the data. Multiplan does not save the password anywhere so the user must remember it. It is required to read the file. If the password is lost, there is no way of reading the encrypted file.

Although multiple windows were allowed in previous versions of Multiplan, V3.0 goes a step further and allows the user to work with up to eight different worksheets at the same time (eight windows active, each with a different worksheet).

A number of improvements/enhancements have been made in the Printing area in Multiplan V3.0. First of all, it is now possible to embed printer escape/control-sequences in cells surrounding text to be printed to change the print style, size of characters, spacing, etc. These escape/control-sequences are unique to your printer. Information on how to accomplish the above is contained in Appendix F of the Microsoft Multiplan V3.0 Manual: Customizing Your Installation. This is what I meant earlier when I said some features are buried. This information should be in the PRINT command section of the Reference Manual, instead of in an Appendix.

Next, Headers and Footers have been added in V3.0. The user defines a header and/or a footer in a range of cells away from the main worksheet area. The starting cell reference is specified when invoking the PRINT HEADING command (this is also used for footers). The header starts on the line immediately after the top margin, and the footer ends on the line immediately preceding the bottom margin. The user can specify what page number to begin numbering from and the type of characters to use as the page numbers: Roman Numeral (upper or lower case), Alphabetic (upper or lower case), or standard Arabic numerals. Date and time can be included in headers and footers, as well as special characters for aligning the text on the printed page.

In V3.0, user-defined currency symbols are available. This allows the user to select up to 9 national currency symbols which includes the US $. If one of these 9 is not appropriate, the user can define his/her own symbol which can be up to 20 characters in length.

With V3.0, the user has a wider range of values for column widths. Under V2.0, the range of column widths was 3–32 characters. In V3.0, the range is 0–64. If the column width is set to 0, the column is not displayed on the screen, but the information in the cells is available to the user.

A DOS shell has been added to V3.0. This allows the user to use the RUN DOS function to execute a DOS command. However, the command that the user executes should NOT remain as a resident program (TSR — terminate but stay resident). If that happens, an error message will be given to the user and Multiplan will terminate with the loss of the active worksheet, unless it was previously saved.

Other new features in V3.0 are Microsoft Windows support, Networking support, and the ability to change the number of lines on your display screen up to 43, if you have an EGA board.

Earlier versions of Multiplan only supported worksheets with up to 255 rows by 63 columns. For most applications in the beginning, this was more than enough. However, as users began to use the spreadsheet more and more, they began to get larger and larger. Soon the 255 × 63 matrix was much too small to be useful. Version 2.0 increased the support to 4095 rows by 255 columns. This is also true of V3.0.

Some Enhanced Commands

The NAME command starting with V2.0 now allows the user to name a macro just as any other cell or range of cells, and to specify a two-letter macro ID. The name and ID are both used in different ways of invoking a macro. The GOTO command is another command expanded due to the addition of macros in V2.0. The user is now allowed to specify the name or a reference (row/column position) of a macro in the GOTO command. In this case, instead of going to the cell with the macro name or reference, the macro at that position is executed.

In V2.0, the QUIT command was enhanced. If any changes are made to the active worksheet and a QUIT is executed before the worksheet is saved, Multiplan asks the user whether the worksheet should be saved or not saved, or whether the QUIT command should be cancelled. This gives the user another chance to save a worksheet, instead of destroying it immediately when the QUIT command is executed.

The FORMAT command added a couple of new subcommands in V3.0 both of which were discussed before and they are the Units-Currency and the Time-Date subcommands. They allow the user to specify the type of data contained in the cell(s) and how the information is to be displayed.

The PRINT command in V3.0 now has the HEADING subcommand which allows the user to specify the cell reference for a header and the cell reference for a footer.

As was discussed earlier, the TRANSFER command in V3.0 now has the IMPORT subcommand for loading an ASCII file which previously was done via a special utility.

Only one new command appears in the list of commands at the bottom of the screen in V3.0 and that is the RUN command with the DOS, AUDIT, and REPORT subcommands, all of which were discussed earlier.

Although not a new command, I just want to make a few comments on the WINDOW PAINT command, however. The Multiplan manual for V3.0 states that this command is for PC, AT, and compatibles only. However, it DOES work for the H/Z-100, as well, but not as stated in the V3.0 manual. It works as stated in the H/Z-100 Multiplan V1.2 update from Zenith. The H/Z-100 supports only 8 colors, while the PC and compatibles support 16 as stated in the V3.0 manual.

Keyboard

The documentation for Multiplan V3.0 is super, except in the area of key commands. The manual has a number of tables showing the key commands that are to work on ALL micros (I call these generic key commands) and those specific to the PC and compatibles. As far as I can tell, all the key commands in the table (both generic and PC specific) work as stated for my Z-160. The on-line HELP documentation contains these same tables and are identical. This was not so with the V2.0 documentation.

As I said above, the generic key commands are supposed to work with ALL micros, and that includes the H/Z-100, which is one of the microcomputers supported by Multiplan. Unfortunately, not all key commands work as stated in the manual; some don't work at all, while some work a little differently than that stated.

In addition to the key command tables for the generic commands and PC specific commands, there are a number of tables in the on-line HELP documentation for the other supported micros, including a table for H/Z-100 specific commands. This H/Z-100 table is not complete. There are some missing specific key commands. Below are a series of tables that show all the key commands, both specific and generic, for the H/Z-100 and the H/Z-100-PC. I also point out where commands don't work and where they work differently. Hopefully, this will be helpful to Multiplan V3.0 Heath/Zenith users.

Back in the first part of this article, I stated that I wanted to cover features not documented. The feature that fits in this category is the Function Keys associated with the various commands that are in the command line. Most, not all, of the commands can be invoked via a Function Key rather than via the command line procedure of highlighting the desired command and pressing RETURN. These Function Keys work only for the PC types and NOT the H/Z-100.

H/Z-100

Move the Cell Pointer

Up	CTRL-E, Up-Arrow
Down	CTRL-X, Down-Arrow
Left	CTRL-S, Left-Arrow
Right	CTRL-D, Right-Arrow
Next Window	CTRL-W, ;, F10
Previous Window	(generic key command does NOT work & NO specific command)
Next Unlocked Cell	CTRL-F, CTRL-J, Line-Feed, F11
Previous Unlocked Cell	(generic key command does NOT work & NO specific command)

Scroll Window

Page Up	CTRL-R followed by CTRL-E, Shift-Up-Arrow, F8, CTRL-F8
Page Down	CTRL-R followed by CTRL-X, Shift-Down-Arrow, F7, CTRL-F7
Page Left	CTRL-R followed by CTRL-S, Shift-Left-Arrow, F6, CTRL-F6
Page Right	CTRL-R followed by CTRL-D, Shift-Right-Arrow, F1, CTRL-F1
Home	CTRL-Q, Home
End	CTRL-Z, F9, CTRL-F9
Home Window	(generic key commands do NOT work as stated but work the same as for "Home" above)

Select and Execute Commands

Cancel	CTRL-C, Esc
Execute Command	Return
Select Next Item On Menu	Space-Bar, Tab
Select Previous Item	CTRL-H, Backspace
Tab to Next Field in Command	CTRL-I, Tab
Tab to Previous Field	(generic key command does NOT work & NO specific command)
Help (main HELP menu)	H (for Help command)
Help (command specific)	HELP, ?
Recalculate	!
Alternate	CTRL-A (followed by two-letter Macro ID)
Range	:

Edit Cells and Commands

Delete Character (at cursor)	CTRL-Y, Delete, Dchr
Backspace (delete character before cursor)	CTRL-H, Backspace
Character Left	CTRL-K, F3
Character Right	CTRL-L, F4
Word Left	CTRL-O, F2
Word Right	CTRL-P, F5
Reference (make absolute)	@, F12
Edit Macro (on or off)	CTRL-V
Single Step Through Macro	CTRL-T
Record Macro (on or off)	CTRL-R followed by CTRL-R
Refresh Links	CTRL-R followed by CTRL-U

Function Keys to Select Commands

(No Function Keys defined for Commands as for PCs)

H/Z-100-PC And PC Compatibles

Move the Cell Pointer

Up	CTRL-E, Up-Arrow
Down	CTRL-X, Down-Arrow
Left	CTRL-S, Left-Arrow
Right	CTRL-D, Right-Arrow
Next Window	CTRL-W, F1, ;
Previous Window	CTRL-R followed by CTRL-W, Shift-F1
Next Unlocked Cell	CTRL-F, F2
Previous Unlocked Cell	CTRL-R followed by CTRL-F, Shift-F2

Scroll Window

Page Up	CTRL-R followed by CTRL-E, PgUp
Page Down	CTRL-R followed by CTRL-X, PgDn
Page Left	CTRL-R followed by CTRL-S, CTRL-Left-Arrow
Page Right	CTRL-R followed by CTRL-D, CTRL-Right-Arrow
Home (R1C1)	CTRL-Q, Home
End	CTRL-Z, End
Home Window (column 1 but top row on current screen)	CTRL-R followed by CTRL-Q, CTRL-Home

Select And Execute Commands

Cancel	CTRL-C, Esc
Execute Command	Return
Select Next Item On Menu	Space-Bar, Tab
Select Previous Item	CTRL-H, Backspace
Tab to Next Field in Command	CTRL-I, Tab
Tab to Previous Field	CTRL-R followed by CTRL-I, Shift-Tab
Help (main HELP page)	ALT-F1, H(for HELP command)
Help (command specific)	?
Recalculate	!, F4
Alternate	CTRL-A, ALT (followed by two-letter Macro ID)
Range	:, F6

Edit Cells And Commands

Delete Character (at cursor)	CTRL-Y, Del
Backspace (delete character before cursor)	CTRL-H, Backspace
Character Left	CTRL-K, F9
Character Right	CTRL-L, F10
Word Left	CTRL-O, F7
Word Right	CTRL-P, F8
Reference (make absolute)	@, F3
Edit Macro (on or off)	CTRL-V, F5 (in Edit Mode)
Single Step Through Macro	CTRL-T, Shift-F5
Record Macro (on or off)	CTRL-R followed by CTRL-R, Shift-F9
Refresh Links	CTRL-R followed by CTRL-U, Shift-F6

Function Keys to Select Commands

SHIFT	
F3	Value
F4	Window
CTRL	
F4	Alpha
F5	Blank
F6	Copy
F7	Delete
F8	Edit
F9	Format
F10	GoTo
ALT	
F1	Help
F2	Insert
F5	Lock
F6	Move
F7	Name
F8	Options
F9	Print
F10	Quit

Continued from Page 46

```
R84 DB 255,128,247,128,226,128,234,128,234,128,234,128,232,128,253,128,255,128
R85 DB  255,128,223,128,223,128,1,128,0,128,222,128,222,128,253,128,255,128
R86 DB 255,128,225,128,224,128,254,128,254,128,224,128,225,128,254,128,255,128
R87 DB 255,128,231,128,227,128,249,128,252,128,249,128,227,128,231,128,255,128
R88 DB 255,128,225,128,224,128,249,128,243,128,249,128,224,128,225,128,255,128
R89 DB 255,128,238,128,228,128,241,128,251,128,241,128,228,128,238,128,255,128
R90 DB  255,128,239,0,230,0,240,128,249,128,243,128,231,128,239,128,255,128
R91 DB 255,128,238,128,236,128,232,128,234,128,226,128,230,128,238,128,255,128
R92 DB  255,128,255,128,247,128,128,128,8,0,127,0,127,0,255,128,255,128
R93 DB  255,128,255,128,255,128,8,0,8,0,255,128,255,128,255,128,255,128
R94 DB  255,128,255,128,127,0,127,0,8,0,128,128,247,128,255,128,255,128
R95 DB  255,128,207,128,159,128,63,128,159,128,207,128,159,128,63,128,255,128
;
; Following codes are for graphic printer characters:
GCHAR
G1  DB  0,0,0,0,28,0,62,0,62,0,28,0,0,0,0,0,0
G2  DB  128,0,192,0,224,0,240,0,248,0,252,0,254,0,255,0,255,128
G3  DB  0,0,0,0,0,0,255,128,255,128,0,0,0,0,0,0,0
G4  DB  24,0,24,0,24,0,24,0,24,0,24,0,24,0,24,0,24,0
G5  DB  24,0,24,0,24,0,255,128,255,128,24,0,24,0,24,0,24,0
G6  DB  24,0,24,0,24,0,31,128,31,128,0,0,0,0,0,0,0
G7  DB  24,0,24,0,24,0,248,0,248,0,0,0,0,0,0,0,0
G8  DB  0,0,0,0,0,0,248,0,248,0,24,0,24,0,24,0,24,0
G9  DB  0,0,0,0,0,0,31,128,31,128,24,0,24,0,24,0,24,0
G10 DB  0,0,0,0,18,0,18,0,58,0,18,0,18,0,0,0,0,0
G11 DB  0,0,8,0,8,0,8,0,42,0,62,0,28,0,8,0,0,0
G12 DB  36,128,73,0,146,0,36,128,73,0,146,0,36,128,73,0,146,0
G13 DB  0,0,0,0,8,0,8,0,42,0,8,0,8,0,0,0,0,0
G14 DB  0,0,0,0,4,0,2,0,127,0,2,0,4,0,0,0,0,0
G15 DB  0,0,0,0,0,0,0,0,15,128,15,128,15,128,15,128,15,128
G16 DB  15,128,15,128,15,128,15,128,0,0,0,0,0,0,0,0,0
G17 DB  240,0,240,0,240,0,240,0,0,0,0,0,0,0,0,0,0
G18 DB  0,0,0,0,0,0,0,0,240,0,240,0,240,0,240,0,240,0
G19 DB   240,0,240,0,240,0,240,0,240,0,240,0,240,0,240,0,240,0
G20 DB  0,0,0,0,0,0,0,0,255,128,255,128,255,128,255,128,255,128
G21 DB  255,0,254,0,252,0,248,0,240,0,224,0,192,0,128,0,0,0
G22 DB  24,0,24,0,24,0,31,128,31,128,24,0,24,0,24,0,24,0
G23 DB  24,0,24,0,24,0,255,128,255,128,0,0,0,0,0,0,0
G24 DB  24,0,24,0,24,0,248,0,248,0,24,0,24,0,24,0,24,0
G25 DB  0,0,0,0,0,0,255,128,255,128,24,0,24,0,24,0,24,0
G26 DB  128,128,193,128,99,0,54,0,28,0,54,0,99,0,193,128,128,128
G27 DB  1,128,3,0,6,0,12,0,24,0,48,0,96,0,192,0,128,0
G28 DB  192,0,96,0,48,0,24,0,12,0,6,0,3,0,1,128,0,128
G29 DB  192,0,192,0,192,0,192,0,192,0,192,0,192,0,192,0,192,0
G30 DB  3,128,3,128,3,128,3,128,3,128,3,128,3,128,3,128,3,128
G31 DB  255,128,255,128,0,0,0,0,0,0,0,0,0,0,0,0,0
G32 DB  0,0,0,0,0,0,0,0,0,0,0,0,0,255,128,255,128
G33 DB  0,0,0,0,48,0,120,0,120,0,127,0,64,0,0,0,0,0
;
; Following codes are for reverse video graphics printer characters:
RVGCHR
V1  DB 255,128,255,128,227,128,193,128,193,128,227,128,255,128,255,128,255,128
V2  DB  127,128,63,128,31,128,15,128,7,128,3,128,1,128,0,128,0,0
V3  DB  255,128,255,128,255,128,0,0,0,0,255,128,255,128,255,128,255,128
V4  DB 231,128,231,128,231,128,231,128,231,128,231,128,231,128,231,128,231,128
V5  DB  231,128,231,128,231,128,0,0,0,0,231,128,231,128,231,128,231,128
V6  DB  231,128,231,128,231,128,224,0,224,0,255,128,255,128,255,128,255,128
V7  DB  231,128,231,128,231,128,7,128,7,128,255,128,255,128,255,128,255,128
V8  DB  255,128,255,128,255,128,7,128,7,128,231,128,231,128,231,128,231,128
V9  DB  255,128,255,128,255,128,224,0,224,0,231,128,231,128,231,128,231,128
V10 DB 255,128,255,128,237,128,237,128,197,128,237,128,237,128,255,128,255,128
V11 DB 255,128,247,128,247,128,247,128,148,128,193,128,227,128,247,128,255,128
V12 DB  85,0,170,128,85,0,170,128,85,0,170,128,85,0,170,128,85,0
V13 DB 255,128,255,128,247,128,247,128,213,128,247,128,247,128,255,128,255,128
V14 DB 255,128,255,128,251,128,253,128,128,128,128,128,253,128,251,128,255,128
V15 DB  255,128,255,128,255,128,255,128,240,0,240,0,240,0,240,0,240,0
V16 DB  240,0,240,0,240,0,240,0,255,128,255,128,255,128,255,128,255,128
V17 DB  15,128,15,128,15,128,15,128,255,128,255,128,255,128,255,128,255,128
V18 DB  255,128,255,128,255,128,255,128,15,128,15,128,15,128,15,128,15,128
V19 DB  15,128,15,128,15,128,15,128,15,128,15,128,15,128,15,128,15,128
V20 DB  255,128,255,128,255,128,255,128,0,0,0,0,0,0,0,0,0,0
V21 DB  0,128,1,128,3,128,7,128,15,128,31,128,63,128,127,128,255,128
V22 DB  231,128,231,128,231,128,224,0,224,0,231,128,231,128,231,128,231,128
V23 DB  231,128,231,128,231,128,0,0,0,0,255,128,255,128,255,128,255,128
V24 DB  231,128,231,128,231,128,7,128,7,128,231,128,231,128,231,128,231,128
V25 DB  255,128,255,128,255,128,0,0,0,0,231,128,231,128,231,128,231,128
V26 DB  127,0,62,0,156,128,201,128,227,128,201,128,156,128,62,0,127,0
V27 DB  254,0,252,128,249,128,243,128,231,128,207,128,159,128,63,128,127,128
V28 DB  63,128,159,128,207,128,231,128,243,128,249,128,252,128,254,0,255,0
V29 DB  63,128,63,128,63,128,63,128,63,128,63,128,63,128,63,128,63,128
V30 DB  252,0,252,0,252,0,252,0,252,0,252,0,252,0,252,0,252,0
V31 DB  0,0,0,0,255,128,255,128,255,128,255,128,255,128,255,128,255,128
V32 DB  255,128,255,128,255,128,255,128,255,128,255,128,255,128,0,0,0,0
V33 DB 255,128,255,128,207,128,135,128,128,128,128,128,191,128,255,128,255,128
;
;
BUFR    DS      6000
        DS      32      ;SPACE FOR STACK
STACK EQU       $       ;PUT STACK HERE
        END     START
```

*

About The Author

Matt Elwood *is a 13 year old going on 21. He has over the past seven years mastered many different phases of computer operations. He takes great pleasure in making PC programs run on the Z-100 using ZPC, such as Harvard Business Graphics and M.1, an artificial intelligence program. He has taught and given presentations on ENABLE, LOTUS, and WordStar. He has programmed in Turbo Pascal, C, BASIC, Assembly, and even tried COBOL. He is familiar with the UNIX operating system and written some small programs in "C" on it. He has impressed members of the Dayton HUG and several members of Zenith Data Systems. Although he normally does not play games, Silent Runner has kept him interested for a few weeks now.*

Z-100 Shoot'um Up

Matt Elwood
1670 N. Laddie Court
Beavercreek, OH 45432

The Z-100 has been around for a few years now. It has been a great business machine along with all of the units that fellow HUGGIES own and love. While there have been many business applications, and some utilities, and even a few games, there hasn't been a good shoot'um up game for the Z-100. The few games that have been written for the Z-100 have been good. Reactor 100 is a good simulation game, Vega Bound is a shoot'um up game and is fun. Airport, a port from the CP/M game is mental and is one that my dad likes. With its great graphics capabilities and good speed, the Z-100 would seem a natural for more games. Unfortunately, most of the shoot'um up graphics games have been written for the PC machines, until recently.

While at the Sixth HUGCON, I came across a set of four games that used and displayed the capabilities of the Z-100 for shoot'um up action. The games are from Audio Visual Entertainment Software of Potsdam, NY. In talking to the authors of the games, they felt the same way about the Z-100 and games. They received Z-100s when they entered Clarkston College. They soon discovered the lack of something to relax with after an evening of study. For this reason, they wrote the four game set that I will discuss. Because my dad was in a good mood, I was able to get him to buy all four games at HUGCON.

A word to the user of Z-100s. These games, especially "Silent Runner" are catchy. Once you get started, you may/will have trouble getting down to the work that you started to do. My father is trying to complete a series of articles for REMark, but many nights he ends up trying to match my scores playing Silent Runner, without luck as of yet. To get the most out of these games I recommend a color monitor. The colors are well done and add to the effect of the games. Use of a joystick or mouse may help, although the Z-100 keypad works OK. The authors have incorporated a joystick driver in the program. The joystick must be located on the parallel port. Instruction for wiring is included in the limited printer documentation that comes with Silent Runner and Starhawk. The other programs have document files on the disk. There are no beeps with the games, so you can play them without disturbing anybody.

Silent Runner

Silent Runner version 2.1 is a game that pits you against a number of monsters while you try to collect the bags of money placed around the board. The action is fast and the graphics of both the man (you) and the monster are very good. Each board that you play on is different and is cleverly numbered with the bricks. Each level becomes more difficult, as can be expected, and you don't have to worry about running out of levels too soon, as there are 99 to start. You can start at level one or any other level. The other levels can be selected by using the 8 or 2 on the keypad to raise or lower the board level, respectively.

When the selected level is displayed, you are positioned on the board. With the first movement, the monster or monsters will appear and will attempt to catch you. The box on the bottom left of the screen displays the money you have collected and the number of lives you have left. You get to be eaten five times before you lose.

The board displays brick horizontal and vertical walls. You and the monster can run across the horizontal surfaces, but cannot climb over the walls unless there is a ladder available. You, but not the monsters, can move across the ropes between brick surfaces or ladders. You can dig holes on each side of you by using the "0" or "ENTER" keys on the keypad. These pits can be used to trap the monster or to move between levels or enter into areas that are completely enclosed. But you have to be careful not to trap yourself in your own holes so you can't get out or else you've already bit the big one. As you move along, you not only have to avoid the monsters, but watch out for hidden brick trap doors that will drop down a level or trap you with no way out but suicide, or brick surfaces that do not show up until you start to run across them.

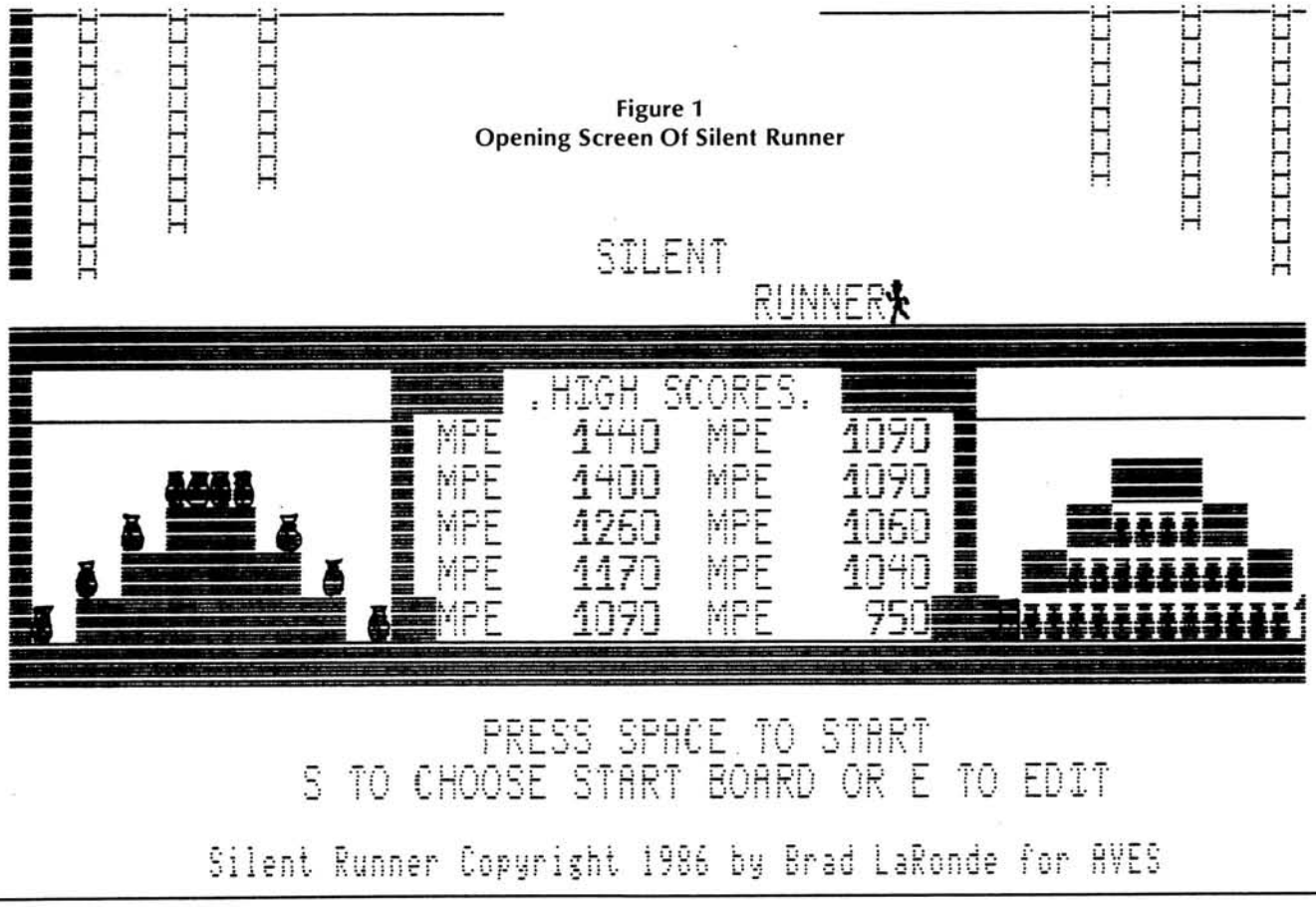


Figure 1
Opening Screen Of Silent Runner

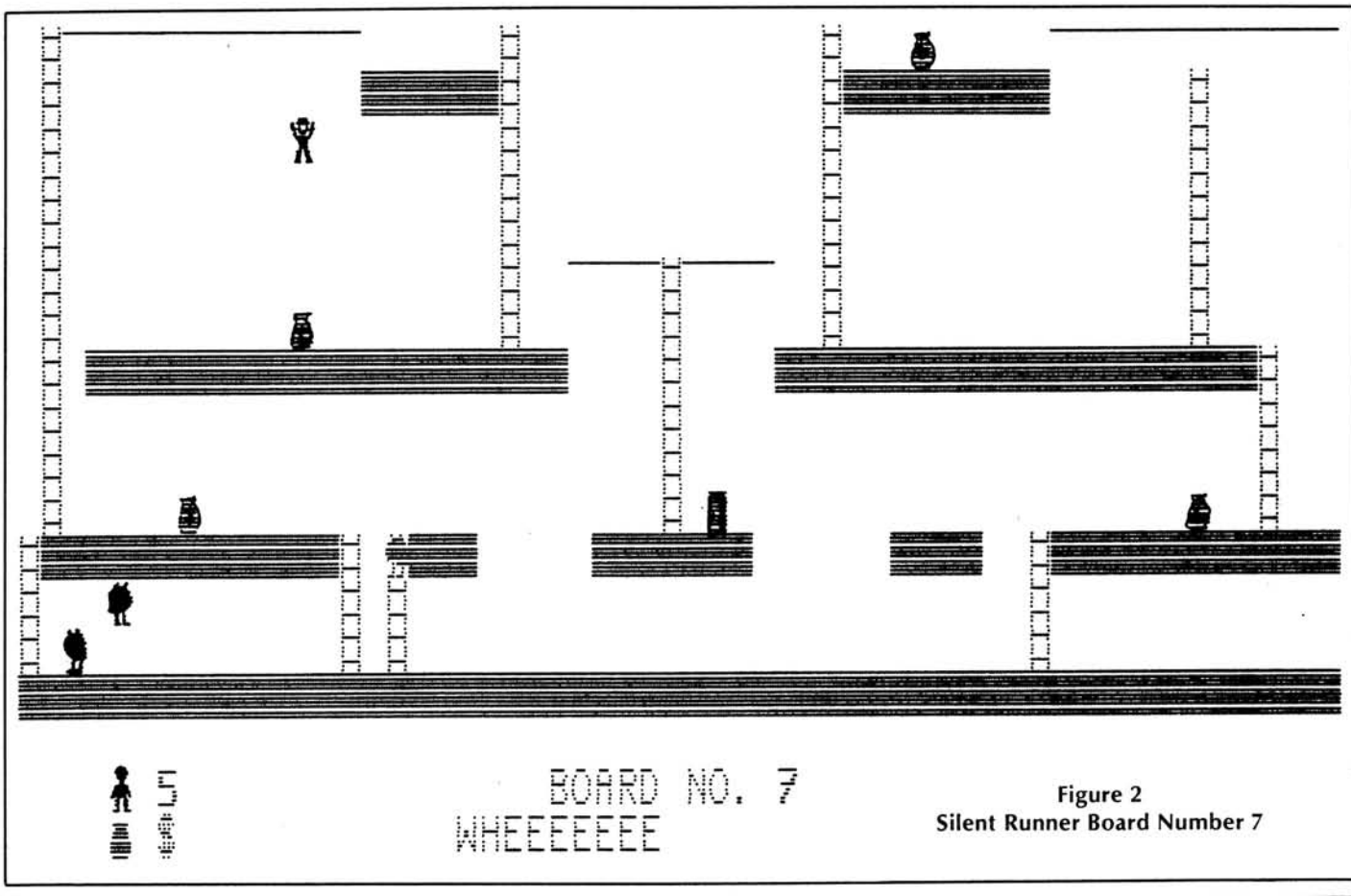


Figure 2
Silent Runner Board Number 7

Running this program on my Dad's (or my) Z-100 with its 8 MHz clock is sporty, but the author thought enough to incorporate a way to slow or speed up the action. By using the "9" or "3" key on the keypad, the action can be sped up or slowed down. The speed is displayed in the upper right-hand corner when you do this. All action is stopped until you hit a direction key. This is very convenient when you are first trying the program and find the monster getting you faster than you can collect the money.

One unique thing that is added to this program is the board editor. Using this function, you can add to current boards or make entirely new boards to challenge you and your friends. I've already made one so hard that I can't even master it. The edit function can be reached by typing "E" at the opening screen. The on screen help makes designing and saving a new board easy.

Airlift

Airlift is an action game where you, as the pilot of a helicopter, are pitted against the bad guys. The goal is to free the POWs that are trapped in structures across the landscape. As you move along through the mountains, the bad guys shoot at you with rockets and cannon shells. You can control the movement of the helicopter by using the keypad. Note the blades on the helicopter as you move along. This is part of the great graphics in the game.

Because you are moving through hostile areas, you are armed with front firing guns and bombs. Luckily, these are provided in unlimited quantities as you tend to use up a lot of them on a mission. The bomb control is the letter "Z" and the forward firing cannon uses the spacebar.

You lift off from your base and move to the right. You move to avoid the rockets and cannon shells, dropping bombs on the gun emplacements and fuel tanks. The graphics from the exploding targets is very good. This would be even better if sound was available. You also drop bombs on the structure holding the POWs. Once this has been set on fire the POWs come out and you can pick them up. You must land and pick up the POWs and then take off without hitting the burning building or mountains. You must land over all of the POWs that you will take back. After picking them up, you must take them back to the starting point. After you land, you will see them walk out of the helicopter and walk to the Operations Building, open the door and go in. You may not be able to pick up all of the POWs from a site in one trip.

The game can be played by two people, with scores for both being displayed on the screen. After you have picked up thirteen of the POWs in the first set of mountains, you move on to another range with increased difficulty. The bad guys add surprises as you increase in difficulty.

As I said earlier, the graphics displays in this game are very good. The two sets of mountains, the foreground and background set are different and move at different speeds. The explosions and resulting fires are well done and continue to "move" throughout the game. The authors used the added capability of the Z-100, and increased the resolution to 640 × 250 so screen dumps would not work on this or the remaining games.

This game is interesting, but does not hold your interest as much as Silent Runner.

Starfire

The third game in the set from AVES is Starfire. This game is basically the same as Airlift, but with a space flavor. You use a spaceship instead of a helicopter, and recover energon cubes instead of POWs, but the action is the same.

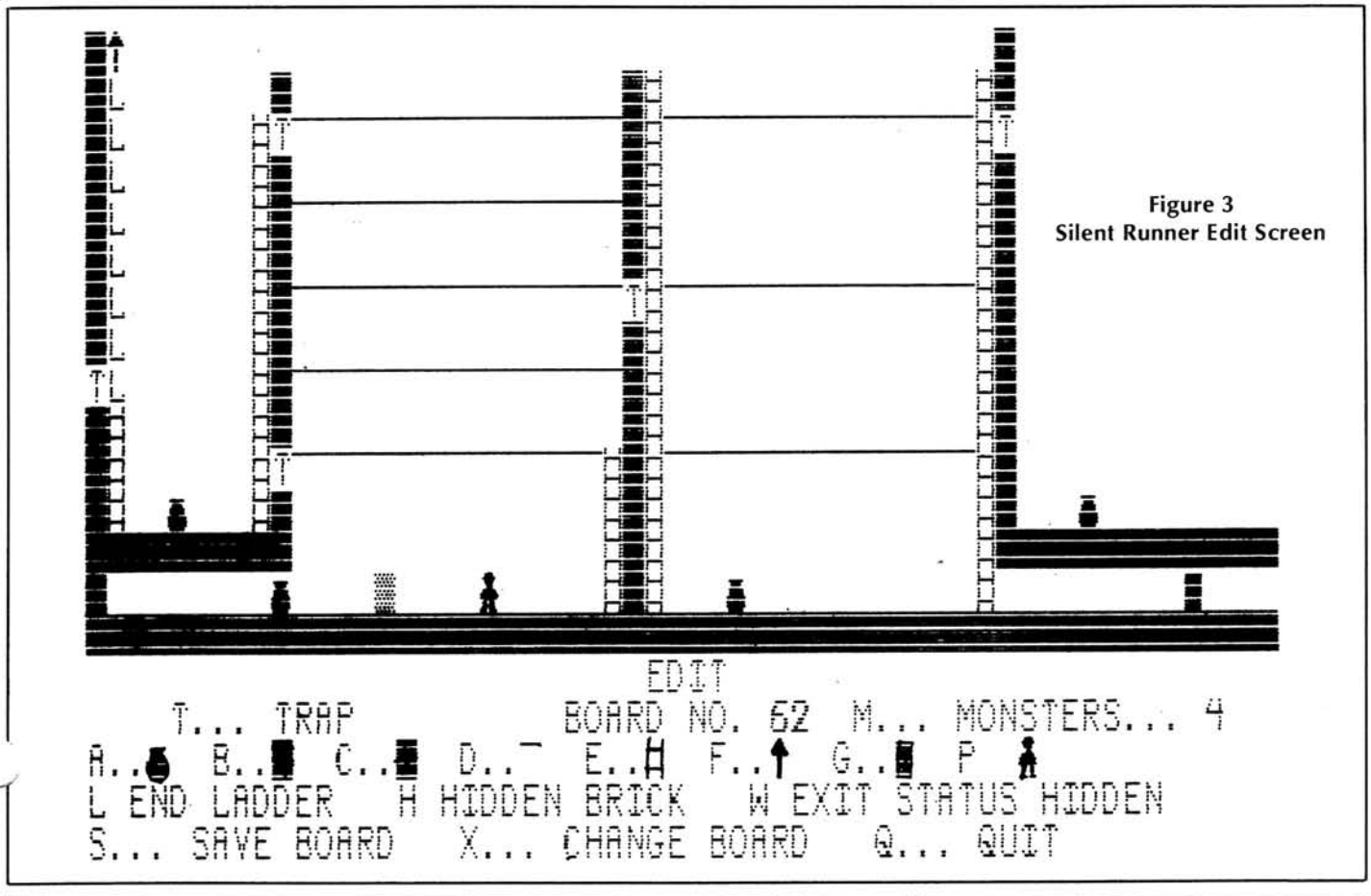


Figure 3
Silent Runner Edit Screen

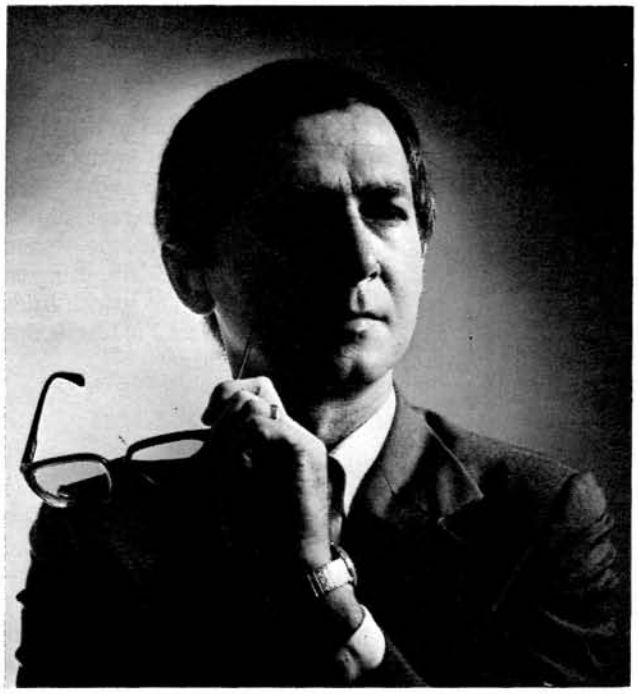
The graphics on the spaceship is very good. It even has a rotating radar antenna. The background is again well done. It has the same two types of mountain ranges, like AIRLIFT, but with a space look. Instead of the sun in the sky, various planets appear as you increase in level of difficulty. You start at Mercury and go to Venus and so on until Pluto, trying to find energon cubes to go to the next planet. Once past Pluto, you've won.

Starhawk

The last game is Starhawk. This game is similar to an arcade game, although not as complex. The spacecrafts even look similar. This game involves you in a battle with alien spacecraft. You have to destroy these crafts by firing your laser cannon. Unfortunately, the cannon has a limited range so you have to be fast. The aliens are also firing rockets as they move down the screen which will keep you on the keypad moving around.

It is extremely hard because of the limited range and the speed. It must be because I can't even get more than 8 of the fighters. Unlike the other games, limits have been set on the amount of laser bursts you can fire, and the amount of time available for each battle. A moving line on the right side of the screen reflects the amount of these items left.

Because I have not mastered this game, I have not moved to the next level. My dad has had better luck with this game and is up to 14 destroyed aliens. As you destroy each level, the alien crafts change. The types are displayed in the opening screen with the credits for the game. The instructions state that after you have destroyed ten alien crafts, you move on to the next level. It also talks about the surface attack. The game is such that a joystick would make it much easier to play. With the keypad, you tend to stay at the bottom of the screen and move back and forth shooting as the aliens come down. *



Mainstream Computing

by Joseph Katz

There's a brief sequel (one might say "a tiny tail") to that true history of The Camden Elephant I related last month. When I showed the manuscript to my neighbor Dr. Strothers Pope for verification, he was so impressed by my accuracy that he told me the rest. Don't leave: it won't take much time.

The Camden cotton farmer had borrowed to finance his super mule. I suppose the bank classified it as an agricultural equipment loan, but Dr. Pope didn't say so and I'm really just speculating. So was the banker whose money floated The Camden Elephant. He lost when it sank. "That's the very last time I'll lend money on anything that eats," he told Dr. Pope. The banker stared at his wingtips while shaking his head ruefully. "Why, after that experience I wouldn't even lend money on Miss America." Which settled the affair of The Camden Elephant once and for all. At least I certainly hope so.

Zenith's new keyboard for the handicapped

An exciting place at the University of South Carolina is the College of Business Administration. Janet's colleague James Finger, Manager of its Center for Academic Computing, and Professor Donald Keating, of the University's College of Engineering, have designed a modified keyboard for Zenith's PC and AT compatible computers that make them usable by handicapped people who must type with a mouth wand. For those people an impenetrable barrier is created by ordinary keyboards, which require them to hold down the CTRL or ALT keys while simultaneously depressing another key. Jim and Don realized that the barrier could be overcome by modifying the keyboard. Zenith has the modified keyboard for special orders. According to Andy Czernek (Zenith's Vice President of Product Marketing) last May, "the modified keyboard will be available at year-end"–which means right now. Nice.

I wish, though, that Zenith would follow the lead of companies like Apple and IBM in supporting work here and at other universities. I don't mean giving discounts to faculty, staff, and students: that's a way to boost sales by turning universities into discount stores. I mean providing equipment and grants, the way other manufacturers do. In my college last year Apple and IBM competed on a proposal to support the teaching of freshman writing. I was special consultant to a committee evaluating those proposals. Zenith's name never even came up. The company simply is not visible here as anything but another minor vendor hustling sales. Pity.

Blitz makes LaserJets and LaserWriters twice as fast

Several months ago, on CompuServe's Aldus Forum, John Tomeny and I started a friendly tussle. John is a talented graphic designer who uses a Macintosh, an Apple LaserWriter, and Aldus's PageMaker. You therefore don't get any points for guessing our battleground.

In the heat of verbal combat, when we were arguing about relative productivity of the two systems, I flatly declared "mine's faster," and as part of my argument I blurted out that I regularly use either my Apple LaserWriter or a Hewlett-Packard LaserJet at 19200 Baud.

"What?" jumped in Joff Morgan, Aldus's System Operator for the forum. "You mean

Address all correspondence to Dr. Joseph Katz, 103 South Edisto Avenue, Columbia, SC 29205. Please enclose a stamped, self-addressed envelope if you want a reply. The volume of mail is too great for me to do more than say "I'll try." Unless your letter says otherwise, it is subject to editing and publication.

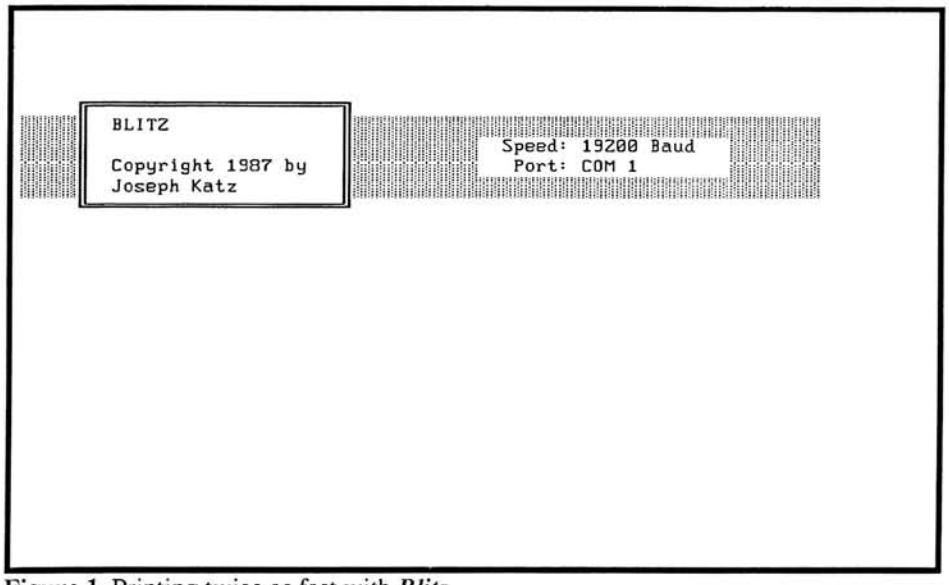


Figure 1. Printing twice as fast with ***Blitz***.

'9600 Baud,' don't you? 19200 Baud is twice as fast as LaserWriters and LaserJets can run. You don't mean you're doing that."

"Yes," I confessed, "I really did mean '19200 Baud'. (See Figure 1 for my commitment.) Yes, I know it's twice as fast as those printers normally run." Since I had let the cat out of the bag anyway, I told Joff that the increased speed is only part of my secret: I suppose you can put an airplane engine in a Volkswagen and have it run for a few miles, but I wanted safety and increased equipment life as well as raw speed. I also wanted reliability: my disposition is not so sweet that I'd be able to tolerate a printer that printed only sometimes, under ideal conditions. My tools have to work for a living, just as I do. I therefore tinkered and tested and spent a lot of time figuring things until I completed the software that gave me my edge. And then I got into the ongoing friendly dispute with John Tomeny, blabbed my secret weapon in desktop publishing, and keeneyed Joff Morgan caught me, "You ought to sell the software instead of sitting on it," Joff said,"because you have something other people need."

So I am. My software package is ***Blitz***. It's all the software needed to run the Apple LaserWriter (and LaserWriter Plus) and Hewlett-Packard LaserJet (and LaserJet Plus) serial printers twice as fast–safely and reliably. It should work with most other serial printers too, but I haven't tested it with them. ***Blitz*** is $50 directly from me. If you want more information, I've done a nifty information sheet I'd be happy to send anyone who sends a stamped, self-addressed envelope. ***Blitz*** works, and I'm proud of it.

If you own one of the Apple LaserWriters and The LaserWriter Connection I did for HUG Software, you'll be happy to know that all the programs in that package (except the 9600 Baud drivers) run superbly under ***Blitz***. I use that combination regularly.

"IBM's Desktop Publisher Lacks Adequate Power"

That was the headline to a September 28, 1987, story in *Infoworld* (p. 5). The story quoted an IBM spokesman explaining that the Model 30 used in IBM's Personal Publishing Solution Pac system is underpowered and might shut down when expansion boards are added.

As I've said repeatedly, the Model 30 doesn't have adequate provision for expansion boards. There are only three slots, and one of them must be used for the board that handles printer i/o and the PostScript interpreter in IBM's system. (Apple's LaserWriter, in contrast, has the interpreter and i/o handler in the printer. Because The LaserWriter Connection and ***Blitz*** mate the computer's i/o with the printer's by a small and efficient software driver in the computer, you don't need special hardware or a slot for it. It's very sensible.) Now it turns out, according to IBM itself, that the Model 30's power supply makes available only about 4 amps at 5 volts to the total three slots available for expansion boards. So IBM's desktop publishing system has its PostScript on a board that requires one of those slots, and it alone draws about 2.4 amps. That leaves a total of about 1.6 amps available to power what you might put into the remaining two slots. "IBM estimates that the average add-in card requires roughly 1.5 amps." Crash!

Although I am not the kind of person to say "I told you so," I have–many times during the past several months. This column was done entirely (except for that photo in the title block) with the system I've designed around my Z-248, using an Apple LaserWriter linked to it with ***Blitz***. The page makeup program I'm testing this time is Ventura Publisher.

The new Zenith Z-171

We bought a new Z-171. I don't mean it's new to Zenith, only that it's new to us. I know the Z-171 has been discontinued. But I got a great buy, and because I need a real portable and couldn't get a Z-183 or similar laptop I thought I'd speculate. I'm glad I did, even though the Z-171 is by no means a perfect machine. It's neither the lightest (14.4 lbs. without the optional internal modem and battery) nor the fastest (only 4.77 MHz, like the original PC); the LCD ("Liquid Crystal Display") is disgustingly slow to refresh, so each screenful oozes down from left to right and top to bottom; the two 360 KB floppy disk drives seem to hold less than when I was a lad graduating from 180 KB single-sided floppies to the new-fangled higher-capacity diskettes; and the keyboard isn't really IBM compatible, especially the equivalents of numeric keypad keys. Nevertheless the Z-171 is a useful machine for a writer or researcher to take travelling, and I see lots of other uses for it besides. If you need a portable and don't truly need the latest and greatest, you might want to hunt for this discontinued machine and use what I've done as suggestions for how to make it work better for you. You can save money that way.

Making portable software

I'm making up a travelling kit of software to carry on trips. At COMDEX I picked up a FieldPro floppy disk case–a padded, zippered case that holds 20 diskettes safely.It was expensive, but the case protects its contents so well that I don't begrudge its cost. After all, if a disk bombs when I'm away from home the trip bombs too. My software requirements are predictable. The programs must be good, of course: silent suffering is not in the Katz family tradition. (Janet bought me a swell button with our family motto on it: "The more you com-

plain, the longer God lets you live.") I also want familiar programs–those I use at home–because I don't want to wrestle with a manual or have the look and feel of a fool who seems never to have used a computer before. There should be none of the fumbling that would take place if I had one set of software on my office machines and a different set for the road. Those two requirements aren't hard to meet. The hard one is that I'm limited to what will work on a dual floppy machine–the Z-171. Many programs tend to become bloated as they move from version to version and the vendors succumb to what we used to call "rampant featuritis." If the trade magazines are right in seeing new popularity in portables and laptops, there'll be profits for manufacturers whose stuff can at least be customized to provide the program's essentials on a floppy disk system.

Here's what I mean. WordStar Professional, Release 4, offers many of the features we now tend to think essential in a complete word processing package: spelling checker, thesaurus, and support for a great many printers, for example. But you can use the WSCHANGE program to strip out most of those features. Then you have the core of WordStar on a single floppy disk, with room left over for the MSDOS operating system kernel and a few useful other programs besides. I have my supple WordStar and PC Outline together on a system disk that also includes my Spin program to speed up floppy disk access and a screen refresh speedup program to help a little with the turgid display on the Z-171. Understand that I like the WordStar spelling checker and thesaurus (it's Word Finder from Microlytics, my favorite), but their presence isn't really essential on the machine I use mostly to bat out stuff while I'm on the road.

It's nice to have a good spreadsheet program handy, and a good database manager too. I wouldn't have thought so, because most of my travelling work requires generating words that eventually will go on paper. But the spreadsheet has come in handy on a couple of occasions in which I had to do some on-the-spot "what if" calculations to see if proposed ventures would pay off. They wouldn't on the initial terms, but Multiplan allowed me to massage one into profitability and showed me that no amount of massaging would make the other venture breathe. If you jettison things like the installation program and the driver for 40-column terminals, the fully-functioning Multiplan 2.0 fits on a single system disk with room to spare for templates. Isn't that interesting? PC-File+ is the database manager I've chosen for travelling. I use it to carry my name-and-address file, of course with phone numbers too. Unfortunately, although PC-File+ itself will fit on on a single floppy, there's no room on that disk for the operating system too. Dangit. Well, at least it's manageable.

I don't see any point in a travelling computer that doesn't allow communications too. Remember that I'm the fellow who can't remember to take along the clothing he packs for his trips. I certainly am not going to trust myself to bring along all the software and data I need to make my trips successful. What I do, therefore, is set up one of my office computers as a host computer before I leave on a trip. (My computers have their own telephone number. And it's unpublished too. No fooling.) That way when I have a late night craving for some bytes all I do is call home and download them. In addition I work with publishers who welcome my articles in WordStar or ASCII files. There's nothing like being able to send a file across the phone lines to those people. Because I can do it on the road, moreover, I don't even have to anticipate my deadlines before I leave. I can string them out as usual and even miss them by the usual few days. That way I don't risk spoiling my editors. And there's CompuServe and other systems I cover regularly. With the Z-171, a communications program, and a modem I can stay in touch while I travel. Of course that means my editors can stay in touch too, but I'm old enough to recognize there are drawbacks to almost everything. Mirror II is the communications program I take with me: everything I need goes on a system floppy. My only problem in this respect is my travelling modem. I don't have Zenith's internal modem for the Z-171, so I've been carrying around my Hayes Smartmodem 1200. That's a nuisance. I'm looking around for a good portable modem for the Z-171. My emphasis is on "good": I'm aware of many portable modems, but I'm also aware that many of them have problems of one or another kind. I'll let you know if I find something likeable.

Touchbase Systems' WorldPort 1200 Modem

I just did. Between the time I finished this column and the deadline for sending it to the magazine, Touchbase Systems' WorldPort 1200 modem arrived. That was yesterday. The name of the modem is bigger than the modem itself, which is about the dimensions of a cigarette package. Of course I haven't had enough time with the WorldPort 1200 to do more than give it a quick test drive. But, also of course, you know I couldn't resist doing at least that much right away. I wanted to see if something so small and would really work.

So I did what you're not supposed to do with new hardware. I ignored the manual, cabled the modem to the computer, plugged the telephone cord into the modem, ran Mirror II and dialed up CompuServe. Sonofagun. This tiny little thing really does work. And it's Hayes compatible too. My script for accessing CompuServe is moderately complex, enough so it can serve as a quick test for everyday Hayes compatiblity.

Mike Bernard of Touchbase Systems had told me all sorts of incredible things about this modem. Its power is either an ordinary 9-volt DC battery inserted in the modem or an external AC adaptor, and the modem senses which is in use. There's no power switch because the WorldPort 1200 also senses when it's in use. (No, I didn't believe that either. But it seems to be true.) It's auto-answer/auto-originate too. It supports the Bell and CCITT communications standards. It works as a direct connect modem or—if you get the optional cups—as an acoustic coupler. It's a 300/1200 Baud modem. What it doesn't have is an internal speaker (but the four LEDS on the box include Call Progress that lets you monitor, accurately, what's happening) or support for 110 Baud (who cares?). I miss hearing the electronic beeps put out by my Hayes and the Zoom Modem when they dial: maybe it's a sign of abnormality, but I keep the volume control on those modems turned all the way up all the time, which irritates Janet when we're in a motel room and she's trying to read while I'm logging on to a system. She applauds the absence of a speaker in the WorldPort 1200.

So far all of those incredible things Mike Bernard told me about the WorldPort 1200 have turned out to be true. And there's more that I haven't had time to try out yet. I think trying them out is going to be fun. The WorldPort 1200 might be the portable modem I've been looking for, maybe even more. I pat myself on the back for having researched the field well enough to have hit modem pay dirt immediately.

Something else. I don't like "dedicated investments"—expensive things that work only with one piece of equipment—which is why I don't spend a lot of money on dashboard car radios and why I don't want a

modem that can be used only with the Z-171. Right after I did some test spins with the WorldPort 1200 on the portable, therefore, I moved the modem to my Tandy DT 100 terminal. That setup works too. Hot dog!

Right now, at any rate, I can go on the road with tools for the basic microcomputer applications of word processing, spreadsheeting, telecommunicating, and databasing. When I contrast this setup with a Radio Shack Model 100 I had to use for a while a couple of years ago, I think I'm in heaven. I hated using the Model 100 because the screen was hard for me to read, the keyboard was too different from any other, and the ROM-resident software was too primitive for my tastes. I like the Z-171, especially since I was able to soup it up a bit with almost no effort and at very little expense.

Souping up the Z-171

Of course the first thing I did was to fill out the machine's RAM from the 256KB with which it is supplied to the 640KB that is the least an MS-DOS microcomputer should have. Fortunately I had supplies of 256 KB RAMs from when I was trying to corner the market a while back and 64 KB RAMs from when I upgraded my Z-100 a few years ago. Installing these was simplicity itself. Unscrew the back of the case, slip off the memory board that piggybacks on the main board, insert the chips, set five jumpers, and reassemble everything. The instructions in Appendix A of the *Z-170 PC Series Portable Computer Owners Manual* are really good enough so the job took only a few minutes, once you figure out the key to the jumper settings.

Next I popped a copy of Zeno, a public domain screen speedup program I wrote about in connection with Microsoft Windows some time ago, on each system disk. Zeno speeds up screen refreshing by about 40% on my machine. The LCD still won't win any prizes for speed. PC-File+ is agonizingly slow, and Mirror II is no speed demon, but Multiplan is okay and WordStar (as you would expect) is better than just okay. It's pretty good. So I have Zeno run automatically, from the AUTOEXEC.BAT file on my system disks, whenever I boot the Z-171.

Then I added the command to run my Spin program so the disk drives would go into action faster. Spin is my version of a program Pat Swayne wrote to reduce the head settle and motor on delays of floppy disk drives that can work faster. The Z-171 drives in my machine need no delays at all, so my AUTOEXEC.BAT command is "SPIN FAST." I included the source code (C and assembly language) in my "C Notes" column in *Sextant* a while back, and I uploaded the executible program to the HUG Forum on CompuServe (GO PCS48).

A low-cost portable power pack

A few days after the Z-171 got here we had to make a weekend business trip to North Carolina and I wanted to use that trip to test drive the machine. Although I didn't really need to run it on batteries then, I do want that kind of portablity available as an option for other trips. It's not that I plan to write on safari or in airport waiting rooms (I avoid anything more dangerous than an outraged reader). Since we drive everywhere, I liked the idea of being able to work while Janet took her turn at the wheel. So this trip I wanted to see what could be done with the Z-171 without a long extension cord. And I didn't want to buy an internal battery for the Z-171. There seemed no real advantage in having so compact a setup when travelling by car (or by train either, which is the other way I go places) and I like to avoid "dedicated investments"–money spent on accessories that can be used with only one machine–whenever possible. So I did a little scheming and came up with a solution that suits me nicely. Maybe you'll be interested too. I think this gadget will work with other portables. It does work with the Z-171.

The Z-171 needs 12.8 volts DC at 2.5 amps which–conveniently and probably not at all coincidentally–is a pretty common requirement for devices like portable television sets, video cassette recorders, cameras, strobe lights, and cellular telephones. Instead of searching out an economical source of power specifically for a portable computer, therefore, I began shopping stores that catered those other kinds of gadgets. There was a plenitude from which to choose. What I eventually chose was Radio Shack's Portable Power Pack (Catalog Number 23-182) at $59.95. Its nominal capacity is 12 volts DC at a comfortable 5 amps. It comes in a handy nylon carrying case with shoulder strap and belt loop, weighs 6lbs., and includes two kinds of rechargers: an AC charger that plugs into any standard wall outlet, and a DC charger that plugs into a car's cigarette lighter outlet. Nice.

Unfortunately the power cable that comes with the Portable Power Pack is wrong for the Z-171. Fortunately it was a simple matter to build the right kind of power cable. At Radio Shack I picked up a Heavy-Duty DC Power Cord (Catalog Number 270-1534B) with a cigarette lighter plug on one end and a coaxial DC plug on the other. The price is $4.99. That coaxial DC plug on the Heavy Duty DC Power Cord is the wrong size for the Z-171 jack, though. I had to snip it off and solder a replacement: 5.5mm outside diameter, 2.5mm inside diameter. Radio Shack sells a pack of two (Catalog Number 274-1573) for $1.19. I don't think I'll need the spare, but I don't mind having it in my junk box. Who knows what awaits around the corner? So my portable power supply for the Z-171 cost me $65.63 plus sales tax and a couple of solder joints. Not bad.

How did it work? Very well. Perhaps too well, considering that my family of computernauts was with me. I spent about an hour writing on the Z-171, starting with one fully-charged power pack and two pair of envious eyes on me. Then the eyes and I went to dinner. After dinner, in our hotel room, Janet slipped into something comfortable: WordStar on the Z-171, which she absolutely needed to use for a project that had a Monday deadline. Of course. An hour and a quarter later she looked up and said, "The computer just told me that the battery is low and I had better save my work to disk. You can take over, dear." Such generosity Well I did take over, just to see what would happen. For the next half hour I got periodic warnings that the battery is low and I should save my work to disk, but nothing bad happened. Then the remaining pair of envious eyes could not be resisted. I quit work, unplugged the power pack, plugged in the AC converter supplied with the Z-171, and shot up some enemy submarines with the assistance of Matthew Katz—who would be of much greater assistance if he stopped trying to elbow me away from the controls of MicroProse's Silent Service submarine warfare game. I'll say more about MicroProse, because I've just discovered they make the kinds of games . . . uh, I mean "simulations" . . . that interest me. Right now I have to say that Matthew wishes I would stop trying to elbow him away from the controls while he's Captain of the sub. Matthew has also declared that he needs a computer of his own now–like a Z-171–to do his school work. "You don't want me to grow up computer illiterate, do you?" Precocious children are a pain..

At any rate, we got better than two and three-quarters of an hour on the very first run of the portable power pack—with, as I have said, a long pause between two sessions. That's not bad, is it?

The best get even better: updates

XyWrite III Plus

You don't know it but I've written this column with XyWrite III Plus. It's an especially good program for use in desktop publishing with the Big Two page makeup software packages, PageMaker and Ventura Publisher. Both of them understand some basic formatting features of XyWrite. Even better, insofar as Ventura Publisher is concerned, it's a snap to transform XyWrite's formatting codes into those understood by the page makeup program and a PostScript printer to do things like en and em dashes. XyWrite is a natural for this kind of thing because its ancestry is in typesetting, specifically in the ATEK system used by many newspapers.

The obvious big news about XyWrite III Plus, version 3.51, is its thesaurus and spelling checker. They are Wordfinder and the new Spellfinder, both from Micrólytics. Wordfinder (which is two words in other products) is my favorite thesaurus program. Spellfinder is not yet my favorite spelling checker, but it's darned good. What you really have to see is how these things work when integrated into XyWrite III Plus. Wordfinder is fast. Spellfinder is not merely fast: it is *instant*. In fact Spellfinder in XyWrite III Plus is so blindingly fast that I always think it's broken the first time I use it after opening a document. What I do, every single time, is give it a nonsense word ("XYZZY" is good) to assure myself that Spellfinder is really working. I don't know what magic Mike Weiner at Micrólytics or the people at XyQuest have wrought, but I am never prepared for how fast this combination works. Even if you're not in the market for a new word processing program, bother someone to give you a demonstration of XyWrite III Plus checking the spelling of a document.

Speed is not the whole story to the way XyWrite's spelling checker works. Since you probably didn't believe me when I said it's "instant," you probably won't believe the rest either. How about a function to help you translate a document from one language to another? How about a function that not only automatically checks spellings as you type but also–get this–can automatically correct them too? (I've just tried it again. It's fun. It's also eerie.) There's more, all equally unbelievable. Get a demonstration, then go tell your friends what you saw and see if they believe you.

Here's a quick list of some other new things in XyWrite III Plus: redlining (for lawyers and, also, textual scholars); document appending; long directory listings that include the first few lines of each file; more ways of inserting time and date in a document; reference tags (for "see also" references, which XyWrite will update as you modify the document); embedded printer control codes and strings; nice directory and subdirectory handling; supplied running header formats; and improved support for terminate and stay resident programs. Many of the other features new to XyWrite III Plus are just as noteworthy. One of the best just got better.

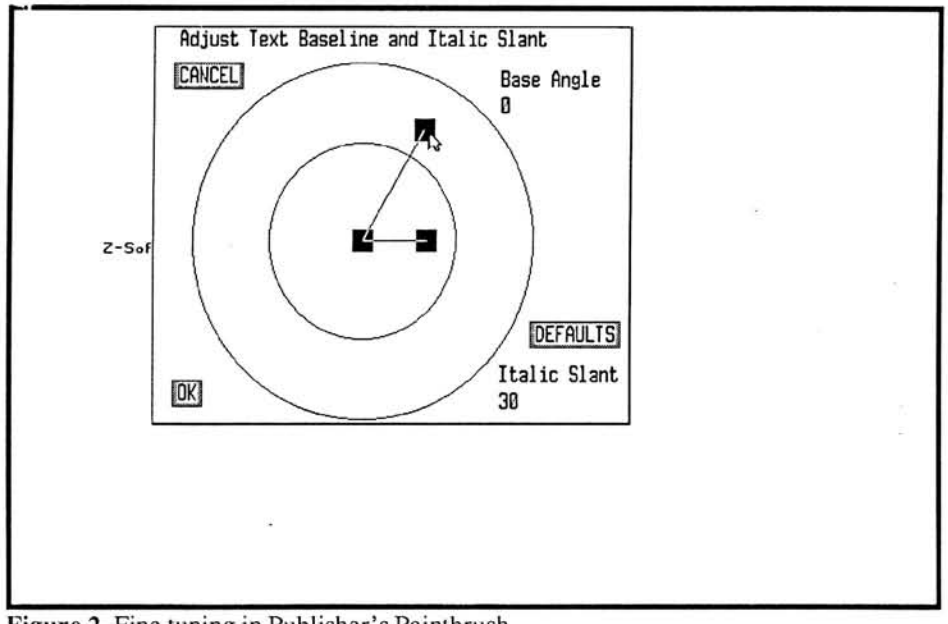


Figure 2. Fine tuning in Publisher's Paintbrush.

Publisher's Paintbrush

Z-Soft's PC Paintbrush, PC Paintbrush Plus, and others in this series of paint programs are the standard by which other paint programs are judged. They are so widely used that their PCC and PCX file formats have become ways to exchange paint graphics among other, often competing, programs.

Now there's Publisher's Paintbrush, a version specially tailored for desktop publishing and laser printers. If your computer has enough RAM, you can print a full letter-size page of graphics on a 300 DPI (Dots Per Inch) laser printer. That's the big news, but there are other things that will endear Publisher's Paintbrush to desktop publishers. For example it has the most sensible ways to adjust the size, spacing, and slant of letterforms I've ever seen in a graphics program: you do it visually by stretching, shrinking, or dragging a tool with your cursor. (Look at Figure 2 to see what I mean.) And, again for example, it supports the most extensive list of printers and display devices I've ever seen in a standalone graphics program, one outside an environment such as Microsoft Windows. The support for IBM's new VGA (Variable Graphics Array) is thorough. Better yet, it's much easier to reinstall Publisher's Paintbrush for different devices than it is to do the same thing in Microsoft Windows. It takes about fifteen seconds in Z-Soft's program instead of twice that many minutes, which is what Microsoft's product requires. It's also easier—so much easier that I use Publisher's Paintbrush as one of my standard tests for hardware such as printers, monitors, and display adaptors. Included in this package, as in all the Z-Soft paint packages, is the Frieze screen capture utility. I find it one of the most versatile graphics tools I own, because it's capable of pulling in screens made with a variety of other graphics programs, such as IBM's PC Storyboard. Putting them in PCC or PCX format liberates them for use in page makeup programs such as PageMaker and Ventura Publisher. If you do desktop publishing seriously you need Publisher's Paintbrush. One of the best just got better.

Timeslips III.

Timeslips was the inexpensive time accounting and billing software I recommended to lawyers, consultants, and anyone else who bills by time and expenses. Afterwards I recived many letters saying that Timeslips outperformed much-more-expensive time and billing packages. I agree, and decided to keep up with it.

If you liked Timeslips you'll love Timeslips III, in part because it's more of every-

thing: 3,400 clients (people you bill), 250 timekeepers (people whose time you bill), and 250 billable activities (such as "Phone call" or "Take abuse" or whatever else you'd like), and lots more space for descriptions of work performed. There are lots more features too, and every one of them is a winner. My favorites are the ability to define macros for oft-performed operations (such as producing bills), and flat fee or minimum or maximum billing as well as billing for actual time and/or expenses. That latter feature makes Timeslips III even better than the earlier version for consultants. One of the best just got better.

PC Quik-Art's The Graphics Link

The Graphics Link is that program I've been using to convert from one paint format to another. Say you get a graphic done in Macpaint on the Macintosh or in GEM Draw on a PC and want to massage it in Publisher's Paintbrush for use in PageMaker. The Graphics Link does the essential format conversions—and back again, if you'd like. It's a useful tool for graphics work, especially for desktop publishing.

So when Federal Express dropped off version 1.50, I gave it an immediate test drive and my jaw dropped. If PC Quik-Art considers this version a minor revision of the program, their standards are much higher than mine. I can't imagine what they would consider a major revision. Version 1.50 is a different program from version 1.00, and better in every way. It does everything the earlier version did, but so much faster that I had to call PC Quik-Art to find out what they had done to the program. What happened was that the programmer rewrote it from Turbo Pascal to the C programming language. Now this useful tool zooms. The Graphics Link is fast enough so I use it now to view uncatalogued paint files, files I want to look at to see what's in them. Version 1.00 had a View option too, but it was kind of slow. Everything was kind of slow in the first version when compared with the way they work in this new version. This new version also has an option to scale paint graphics. You can, for example, specify enlargements and reductions by percentages. Scaling is important because paint graphics are distorted when stretched or shrunk from their original size. Scale them instead. I'm impatient to see what PC Quik-Art considers a "major" revision. Notice, by the way, that PC Quik-Art has a toll-free number for tech support as well as orders, and that an owner of 1.00 can upgrade to 1.50 for only $5. One of the best just got better—at only nominal cost to you

See you later.

Products Mentioned by Joseph Katz

Blitz. Ver.1.0. $50.
Joseph Katz
103 South Edisto Avenue
Columbia, SC 29205

WorldPort 1200 Modem. $199.
Touchbase Systems, Inc.
16 Green Acre Lane
Northport, NY 11768
516/261-0423

XyWrite III Plus. Ver. 3.51.
XyQuest, Inc.
Box 372
Bedford, MA 01730
617/275-4439

Publisher's Paintbrush. Ver. 1.03. $285.
Z-Soft Corp.
1950 Spectrum Circle
Marietta, GA 30067
404/980-1950

Timeslips III. $199.95.
North Edge Software Corp.
Box 286
Hamilton, MA 01936-9986
617/468-7358

The Graphics Link. Ver. 1.50. $99.95
(Upgrade from 1.00: $5)
PC Quik-Art
394 S. Milledge Avenue
Athens, GA 30606
800/523-1796

*

Printing Blocks With WordStar 4

Pat Swayne
HUG Software Engineer

In my article, "WordStar Versus WordStar" that appeared in the October 1987 REMark, I mentioned a trick that I sometimes used with WordStar release 3.3 for printing small portions of a file. I would mark the part I wanted to print as a block (using the block marking commands, ^KB (Control-K, B) and ^KK), and then use the block write command (^KW) and specify PRN as the name of the file to write. This would cause the marked portion to be printed.

As I stated in the October article, this trick does not work with the new WordStar release 4.0, but I have since worked out a patch that lets the trick work. You can make the patch using DEBUG by following these steps:

```
A>REN WS.EXE WS.BIN
A>DEBUG WS.BIN
-E22E2
xxxx:22E2   72.90    04.90
-W
Writing xxxxx bytes
-Q
A>REN WS.BIN WS.EXE
```

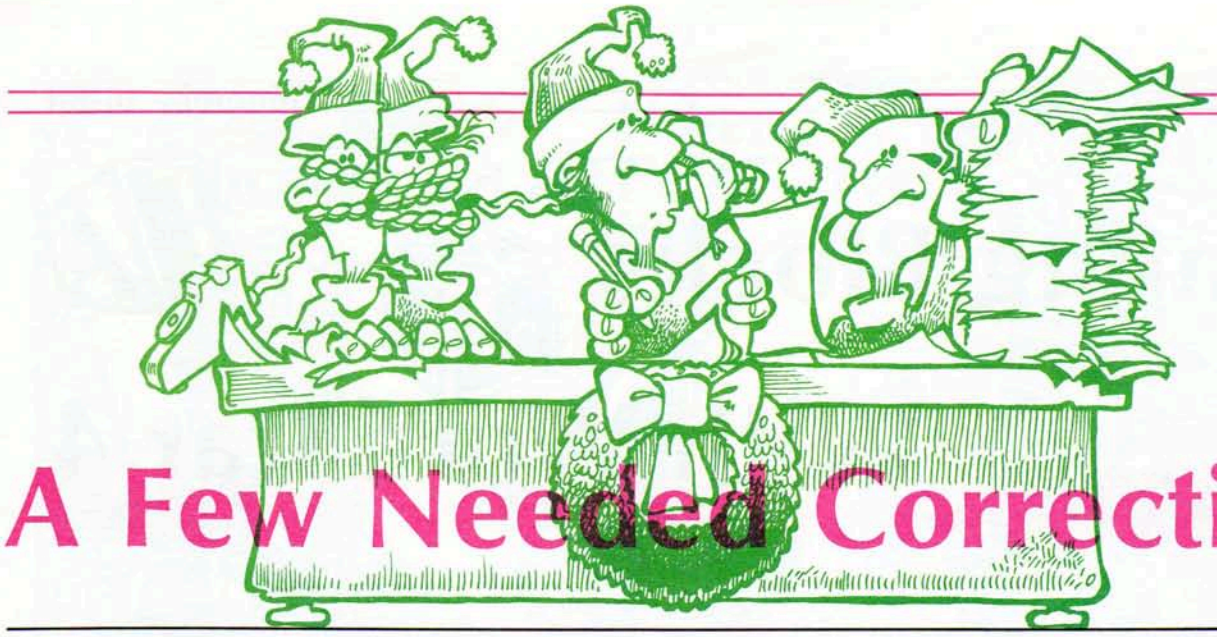
In the above example, WS.EXE and DEBUG are both on drive A:. The x's refer to numbers that are not important for this example. After making the patch, the trick of writing a block to PRN will work, but when you give the command, a message will appear that says, "That file already exists. Overwrite? (Y/N)?". You must type Y, and then the printing will start. You will then get another message that starts, "Can't create a file.", but only after the information has been successfully sent to the printer. You must then press Escape to return to normal editing.

This patch removes the code that returns an error if an attempt to delete an old file is unsuccessful. If you use ^KW to write to an actual file, and a file of the same name already exists on the disk, you are first presented with the "file already exists" message, and when you type Y, the file is deleted. If you use PRN as your file, MS-DOS will indicate that PRN is a file that already exists and WordStar will attempt to delete it. Since PRN cannot be deleted, the attempt will return an error, and that is where WordStar gives up the attempt to print a block without my patch.

This patch does not affect the use of ^KW with actual files, since if the attempt to delete a file is not successful, the attempt to write over it will also be unsuccessful, and WordStar will give you an appropriate error at that point.

It is hoped that MicroPro, the creators of WordStar, will take note of this problem and fully restore the ability to write blocks to the printer in future versions.

*

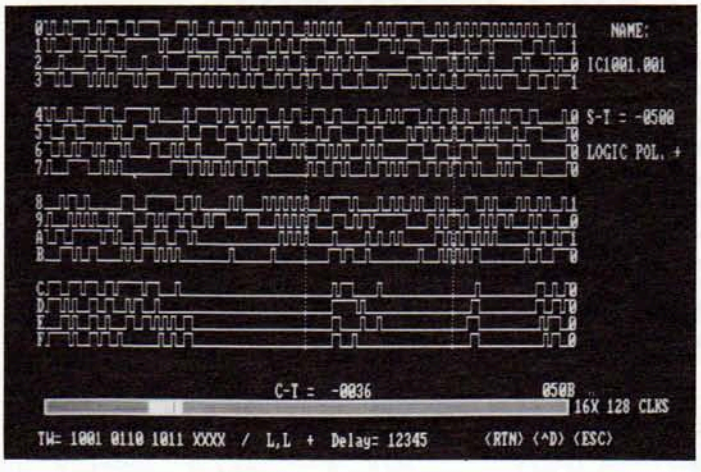


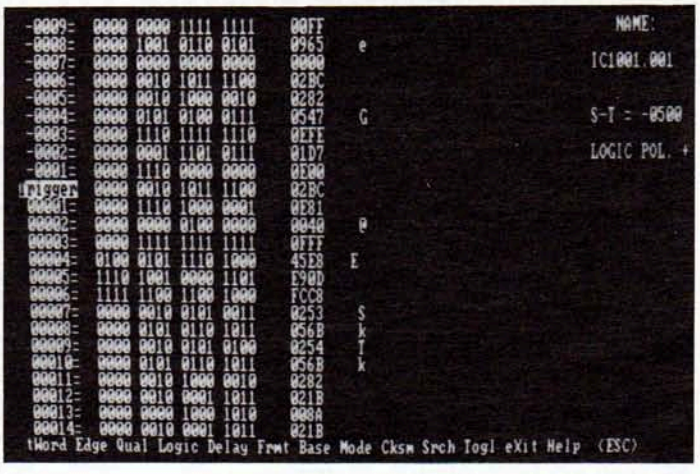
A Few Needed Corrections

The Heath IC-1001 Logic Analyzer

Terry Perdue
Heath Design Engineer

These photos were inadvertently left out of Terry Perdue's article that began on Page 21 of the November 1987 issue of REMark. On the second page of the article, first column, last paragraph, it says "see photos" in parentheses. These are the photos you should have referred to. Sorry for the oversight and any inconvenience this may have caused.





November 1987 Table of Contents

On Page 4 of the November 1987 "Table of Contents", the article entitled *"C/80 — Eliminate Library Duplication"* was not done by William M. Adney and was not on Page 29. This article began on Page 23 and was written by Dave O'Meara. William M. Adney's article entitled *"On The Leading Edge"* was on Page 29 and was not listed in the "Table of Contents". We're sorry about this mix-up, and we hope everyone was able to find these articles.

SPREADSHEET/DATABASE Corner III

H. W. Bauman
493 Calle Amigo
San Clemente, CA 92672

There are two commands that MUST be used for almost any database project! Can you name them before going on? They are:

1. SORT
2. INDEX

All database software have some kind of SORT command. Ashton Tates' dBASEII, dBASEIII, and dBASEIIIPLUS also have a very important INDEX command. I will point out some other software that have commands that do similar functions in future articles. I am using these programs because more readers have told me that this is what they have. (It does not mean that I am saying it is the best, which it was at one time, but NOW other companies have come out with competing software.)

When the user comes to use the database file or files, it is almost all the time that the files must be arranged in some definite form, like alphabetic or numeric! Both of the above commands will do this. However, as I will be demonstrating, the INDEX function has many advantages.

If the reader has completed the past two projects, two database files should be on the data disk! If the user does not have these two files, STOP now and do these projects! You will not be able to do this project or understand it. It is necessary TO DO to understand! Do you agree? The two database files are named CLIENTS and CLIENT1. Before proceeding, if you do not have these two files on the data disk that you are using, transfer them to your current disk and have this disk in the drive that you have selected as the data file drive.

Now, at this point, I would suggest that the reader look at the two above named commands in the manual, and also check the HELP screens for all the information that is available.

The user will find that when the SORT command is used to arrange the database files, dBASE requires the program to create a new data file to store the rearranged sorted files on the data disk in a file called TEMP. Once the SORT is complete, leave dBASE and run a DIR on the data disk and check for the TEMP file. Now, go back into the dBASE program and use the COPY command to put the contents of the new sorted file called TEMP back into the original file!

Now, let's try using this procedure:

1. USE x:CLIENTS
2. LIST (Just to check that we have the right file)
3. Print the file
4. SORT ON LNAME TO x:TEMP
5. LIST and Print it
6. HEY! It did NOT SORT the records!
7. Do you know why? We just did LIST on CLIENTS over!
8. USE x:TEMP
9. LIST and Print it. Remember sorted files are here!
10. COPY TO x:CLIENTS — CLIENTS already exists — Y
11. Remember, we want CLIENTS to be active file
12. LIST and Print it
13. Use DIR x: to check what files are on the disk.
14. Did TEMP show? After the COPY, we know that two database files — CLIENTS & TEMP — are the same. Use DOS DELETE to remove TEMP. Oh, did you note size?
15. What needs to be done to make a BULK MAILING file?
16. SORT ON ZIP TO x:DMAIL
17. USE x:MAIL
18. LIST and Print it
19. Records should be in Zip Code order, ascending. What if we wanted descending?
20. SORT ON ZIP/D TO x:DMAIL
21. LIST and Print it. Did it work?
22. (/D) specifies descending order!
23. What can we conclude from the use of the SORT command? One, it doubles the disk space used. Second, the SORT process is very slow! Think if your file had 1000 entries. Third, the Record Numbers have become rearranged. Wouldn't it be nice if the

Record Numbers were to remain fixed?

24. Let's see if dBASE has a better procedure!

The dBASE INDEX command provides a much faster method, and has a big disk file space saving method of sorting records! The command syntax is similar, but the results and approach are altogether different. Trying INDEX procedure is the best way to find the changes:

1. USE x:CLIENTS
2. LIST and Print it
3. INDEX ON LNAME TO x:NAMES
4. USE x:NAMES
5. LIST and Print it
6. Records have changed to the correct order
7. Record Numbers have remained the same
8. This is a helpful file, LNAME still has the same numbers
9. NOTE, we do not need COPY TO command
10. INDEX command asked dBASE to INDEX to the new x:NAMES file. Note! x:NAMES is not a database file! It is a special file that we will call an "index file".
11. Use DOS DIR:x. What do you find? Do you see x:NAMES.NDX? Notice how small it is!
12. INDEX ON ZIP TO x:ZIPMAIL
13. LIST and print it. What did you find?
14. Let's do the following: USE x:CLIENTS INDEX x:NAMES
15. LIST and Print it
16. When the user wants a quick look at the mailing list in Zip Code order, no re-sort is required since the indexed ZIP and x:ZIP.NDX data is on the x:disk. The user tells dBASE: USE x:CLIENT INDEX x:ZIP
17. LIST and Print it

Not only does ZIP.NDX file take less disk space than the previous sorted file, but the INDEX command serves the same sorting function. In fact, INDEX does it FASTER and has other better features! The beauty of the INDEX command is that it allows new records to be added or changed in a file and the records can be viewed without re-sorting after the changes or going through another INDEX TO procedure again. It is IMPORTANT to remember that this works with only 'active' files. If the user wants to display data like LNAME and ZIP often, type:

```
USE x:CLIENTS INDEX x:NAMES, x:ZIP
```

Now using the APPEND command, new records can be added without problems. After the new records are added or changed, type LIST and view print the records. They will be arranged alphabetically by LNAME. There is no need to use the INDEX ON procedure again. Let's view the records in zip code order by typing:

```
USE x:CLIENTS INDEX x:ZIP, x:NAMES
```

Now LIST and Print it.

WARNING! The user must remember to make the indexed files active, as follows, before adding or changing any records. If this step is neglected, the indexed files will become corrupted. They will have to be recreated. You will know that this has happened to an indexed file by:

A. dBASE will not display the added or changed records when the LIST command is used.
B. dBASE will not display the Error Message — RECORD OUT OF RANGE — when LIST is used.

The corrupted files can be recreated. Two methods can be used. First, use the database without the indexed files with INDEX ON, like this:

```
USE x:CLIENTS
INDEX ON LNAME TO x:NAME
INDEX ON ZIP TO x:ZIPMAIL
```

The second way, use the database file with the existing indexed files as follows:

```
USE x:CLIENTS INDEX x:NAME, x:ZIPMAIL
REINDEX    (new command that should be
            learned)
```

The dBASEIII software permits seven active index files at a time. I would wait until you have experimented with these examples before going beyond a couple of index files. Remember, you will not learn without practice! Work with the many possible indexes with the original x:CLIENTS and x:CLIENT1 files. DO NOT worry about experimenting. It is impossible! Try to find uses for making index files active with the USE command, and then the REINDEX command.

Conclusion

The next article will show how to do Editing and Modifying of database files. Please review the following commands to prepare for this article:

EDIT
BROWSE
CHANGE
REPLACE

After reviewing these commands, write down the MAJOR differences between these commands! Also, if you have the time, look up and review the following commands:

DELETE
PACK
RECALL
MODIFY STRUCTURE

These commands MUST be learned before we can create and print formatted reports. Then, we will try to use the REPORT FORM command!

Paul Mace Software — Utilities

I did not receive the Mace Utilities in time to include them in the Review #3 article. I think that these utilities are really good ones; so, I will add a short review about them here. I received the following:

H-TEST — VERSION 1.5
Mace VERSION 4.10
1. Recovery Utilities — Ver. 4.10
2. Hot-Rod Utilities — Ver. 4.10
3. dBase Recovery — Ver. 1.00

The H-TEST is shipped alone, while the other three are grouped together as a combined package.

As a follow up of Review #3, I started with H-TEST. Paul Mace distributes this package for Kolod Research, Inc. The utilities included in this package are:

1. H-TEST.COM — a hard disk very thorough test program.
2. HFORMAT.COM — a better than DOS hard disk format.
3. HOPTIMUM.COM — a program that will tell the optimum interleave for your hard disk system.
4. RELOCATION.COM — a program to relocate the hard disk head to a desired spot.
5. GETSEC.COM — a head relocation program.
6. PUTSEC.COM — an Absolute disk-(ette) read program.

First, I am sure that the reader will see that this is not a beginner's utility disk. However, we must all get our feet wet at some point. The manuals with this package are an advanced text to try and study from. Do not start with a hard disk, but try a diskette with any backed-up valuable files and experiment. You will find it interesting. I am wiser for trying!

Second, when using this program, DO NOT have any TRS programs on the disk you are testing. They will only get you confused. AGAIN, be sure to have all of the disk programs or data backed up! HTEST with the write test option or HFORMAT will DESTROY all data on the disk! Also, all disk tests are subject to temperature variations. Be sure to run the tests with the drive at room temperature and then warm it up

with a minimum of 2 hours of run time. Be sure that the test drive is mounted on a FIRM, STEADY mount. If the drive to be tested has been pre-formatted, start with HTEST. It will quickly scan across the entire disk and display the bad tracks found.

HTEST is the hard disk program that can be run with various options. A non-destructive (read) or a destructive (write) mode can be chosen by the user. If when running HTEST it encounters massive amounts of faults, the disk most likely has a poor format.

HFORMAT also has many options that can be chosen by the user. With massive errors, it is best to run HFORMAT without options. It will make sure that the controller board and the drive are compatible. (Remember that this utility will not correct a damaged drive. Only Disk Technician has a chance to do this within its 32Mbyte limit in the present version.) HTEST and HFORMAT will provide the user with the maximum usable tracks which can be used. This usually will solve most disk problems. This program will provide a means of moving the Directory and/or FAT tracks to GOOD tracks!

HOPTIMUM is a program that I have been looking for! It is designed to provide the user with the maximum throughput by finding the optimum interleave for the user's system. It will run tests on say 10 cylinders (user can choose more or less) located at some free cylinders on the disk. The utility will test the throughput using an interleave from 1 up through any reasonable upper limit you choose. I tried 10. At the end of the test, it will display a table of interleave vs data handled. From this table, you can choose the optimum interleave for the system. I ran the test using my H-248 running at 8Mhz, 0 wait states with a 20MB full high 39ms full height disk drive. The table showed clearly that 2 was my optimum interleave!

I could write a book on this utility. When H-TEST arrived I worked 6 hours non-stop with it before I could put it down. It is a utility that any serious computer user should have!

Mace V.4.10 Utility package has many very useful programs that work with both diskettes and hard disks, with a manual that gives an education on disk drives. I have become a somewhat better disk user because of my understanding of these utilities. As stated before, this package comes in three volumes:

1. Mace Utility-Recovery Manual. There are many programs to recover lost or damaged files. I would rate this one at the top, if you start with Mace files on your disk at the beginning. If you get "General Failure reading drive C:" message, caused by damage or destruction of the Boot sector or FAT/Directory tracks of the disk, you still have real hope of saving the drive's data.

Restart the computer with the original DOS disk in the A: drive. This MUST be the same version used to format the damaged disk. (Zenith users must hold the ESCape key down until the computer beeps, then wait a few seconds and let the ESC key go. This should put the user in the Monitor, which will show its screen. Type 'bf1' and press enter to boot from drive a:). Put the "RED" RECOVERY disk in drive a:. If it is a hard disk problem, type FORMAT__H C: and press Enter. (In case it is floppy drive B: fault, type HFORMAT__F B:/r and press Enter. The '/r' option means RESURRECT, because most problems are cured with this utility. This '/r' causes the disk data to be lifted, stored in memory, and the media reformatted. If the media tests good, the data is replaced on the disk.) If this fails, Mace has TECH support that can be called at (503) 488-0224.

Now, what is the best way to use MACE? The good news is that this type of failure can be prevented! Replace the DOS format with FORMAT__H.COM or FORMAT__F .COM. Use the "RED" diskette and type UNFORMAT and press Enter. The program will ask which drive. Press the letter of the desired drive. It wants to know if RXBAK file has ever been executed to create and update the file BACKUP.M__U for that drive. If MACE were installed, the message BACKUP.M__U would appear on the screen and the user enters 'Y'. The utility will search the disk for BACKUP.M__U. The program will display — Error Reading Disk ... continuing until it finds a bad spot on the media. This is normal because nearly all disks have some faults. The more faults the disk may have, the longer the test will take. RXBAK creates two copies of the backup info. One will be the current backup and the other will be the previous backup. This is needed where the FAT or Directory media is bad. The new BACKUP.M__U file fills up with bad data. The OLDBACK.M__U will contain the previous and probably correct data. Restoration is automatic. Now run CHKDSK. Errors, if any, would be the result of work done since the last Boot. This would be corrected by running CHKDSK/F several times. If all this does not work, do a FORMAT and then UNFORMAT. Next, do an UNDELETE using the first letter in the file name. Recovery will be automatic. Almost all data loss situations that do not involve media surface damage or physical damage to the heads are recoverable.

I can only touch on the Mace Utilities. Time and space in REMark limits my coverage. I just wanted to go through a typical problem so the user will see how the prompts help. If the user is willing to learn the utilities by working with them, then the user will find many things that can be done. If the user works with computers long enough, the user will find that help is needed and can only be found with the various utilities that are available! There are several makes of utilities on the market and they all overlap to some extent. The serious user will find that more than one is required.

2. Mace Utilities-Hot-Rod Manual. This group of utilities covers many types of problems. Some of these are similar to those I have covered in the Review #3 article. Here is a list of what is included:

A. UnFRAGMENT — All hard disk users should be using one of the various makes available. These provide faster disk access and are a big help in recovering a lost file.
B. VCACHE — This subject will take up most of an article to cover the various methods available. This utility is important because it is a way to increase throughput!
C. SORTD — It is included in many packages.
D. SQZD — A handy utility when your storage disk is filling up.
E. VSCREEN — This utility will speed up the screen display.
F. VKETTE — This is a Caching software for diskettes.
G. Many other interesting utilities.

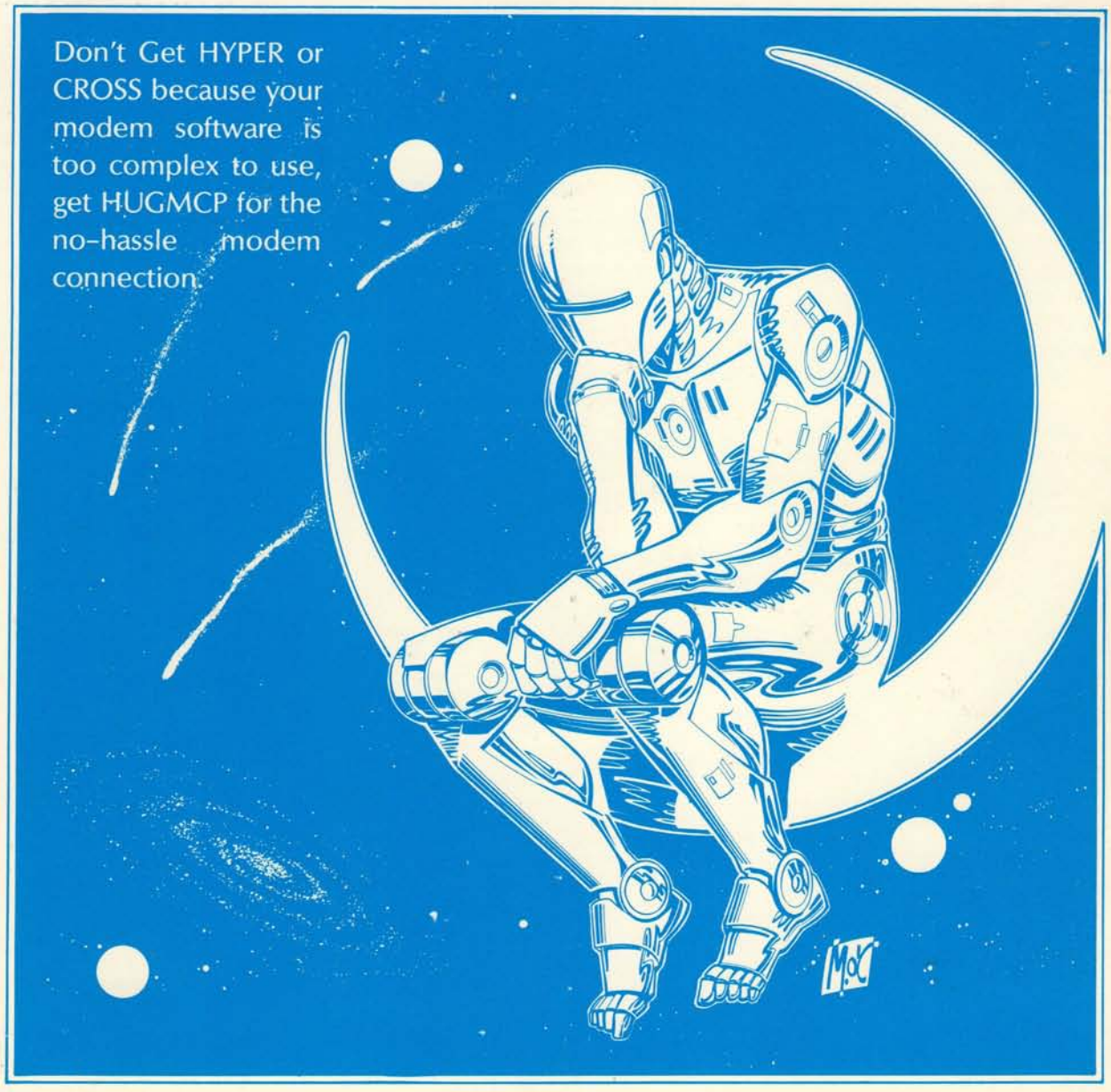
3. Mace Utility — dbFIX Manual. This is an excellent dbase file recovery system that repairs damaged or lost database files automatically. If any user is working with a large database file, this utility is a MUST! This is a specialized problem and I will not cover it. It works on my database!

Conclusion

All the Mace Utilities are very complete and do the job thoroughly. The manuals are an education to read. They must be used while reading them to get the most from them. The computer field is not easy if the user wants to learn!

Paul Mace Software, Inc.
Bill Templeman (Order Direct)
400 Williamson Way
Ashland, OR 97520
(503) 488-0224
(800) 523-0258 Orders only

MACE 4.10 (3 Utility Groups)	$99.00
H-TEST	$89.95 ✱



Heath/Zenith Users' Group

Hilltop Road
Saint Joseph, Michigan 49085

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

POSTMASTER: If undeliverable, please do not return.

P/N 885-2095