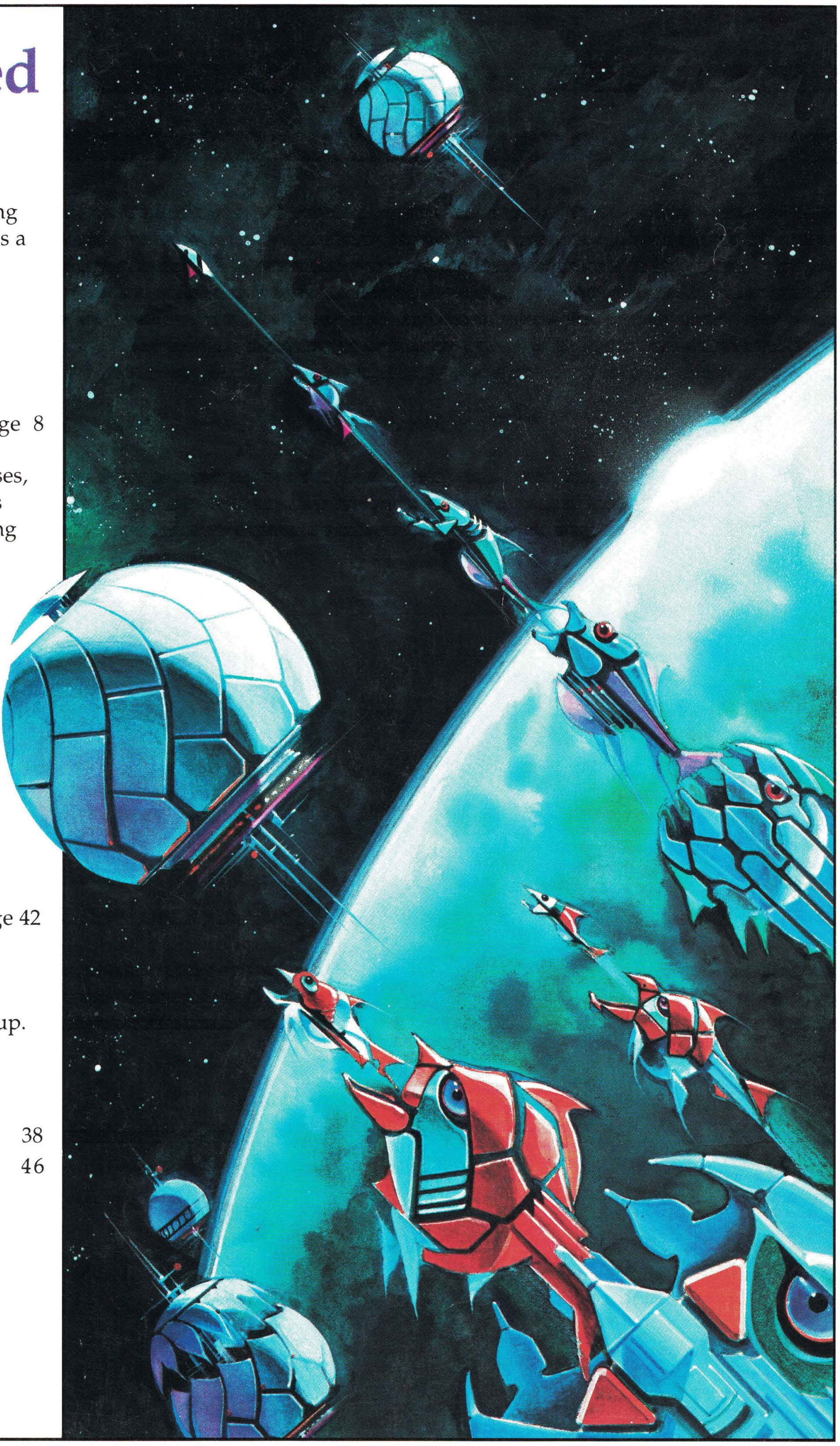
No. 44 November-December 1988 $3.95

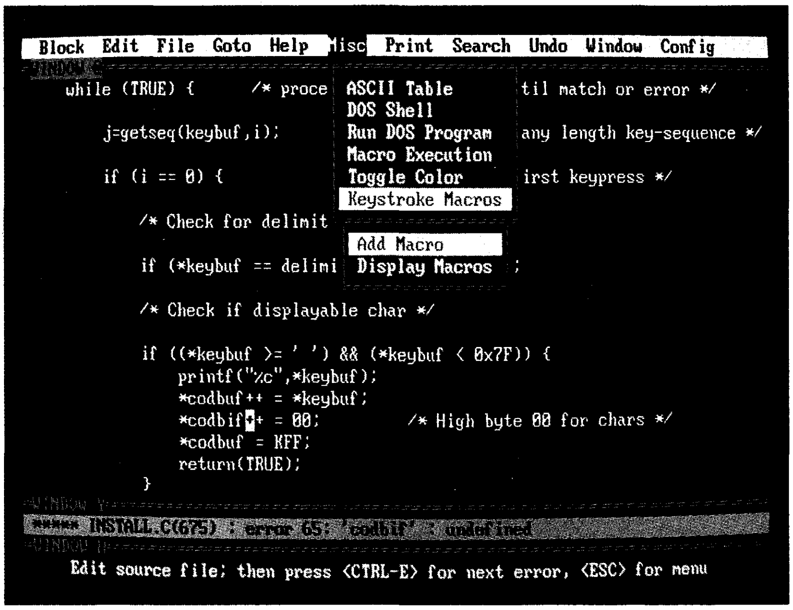
THE MICRO TECHNICAL JOURNAL

MICRO CORNUCOPIA

Object-Oriented Programming

Objects are the latest buzz word. For a long time there was only Smalltalk, now there's a whole troup of players including Actor, Objective C and C++. So we start out this issue with a four-piece trilogy on objects: complete with theory, examples, and projects. (Space is no object.)





```
    while (TRUE) {      /* proce                    til match or error */

        j=getseq(keybuf,i);                         any length key-sequence */

        if (i == 0) {                               irst keypress */

            /* Check for delimit

            if (*keybuf == delimi                 ;

            /* Check if displayable char */

            if ((*keybuf >= ' ') && (*keybuf < 0x7F)) {
                printf("%c",*keybuf);
                *codbuf++ = *keybuf;
                *codbif++ = 00;           /* High byte 00 for chars */
                *codbuf = KFF;
                return(TRUE);
            }
***** INSTALL.C(675) : error 65: 'codbif' : undefined
    Edit source file; then press <CTRL-E> for next error, <ESC> for menu
```

VEDIT PLUS

#1 PROGRAMMABLE EDITOR

NEW VERSION 3.0

- Best Multi-Level Undo
- Regular Expressions
- Pop-Up ASCII Table
- Pull-Down Menus
- Compiler Support
- Column Blocks

Until now, if you wanted the best Undo, the best compiler support, regular expressions and column blocks you chose BRIEF™. If you wanted unlimited keystroke macros, the best configurability, "off the cuff" command language macros and blazing speed, you chose VEDIT PLUS.®

Now the Choice is Easy

The all new VEDIT PLUS 3.0 gives you the best Undo of any editor, the best compiler support, unequaled windows, true regular expressions and extensive new features. We're leading the way with easy to use pull down menus, context sensitive help, a pop-up ASCII table, new printing options and much more. Incredibly, VEDIT PLUS 3.0 is now twice as fast as before and, at only 60K in size, it loads fast!

Completely Configurable

Change a few keys or redefine the entire keyboard, VEDIT PLUS adjusts to your editing style in minutes. You can even create new editing functions using simple keystroke macros or fine tune existing ones. VEDIT PLUS is so configurable that it easily emulates other editors and word processors (WordStar and Word Perfect emulation included). Quickly access editing functions with a single key or through the pull-down menus.

Try before You Buy

We challenge you to experience the dazzling performance and exceptional features that make VEDIT PLUS the best choice. Our evaluation disk includes the complete editor.* Learn VEDIT PLUS using our extensive "training" macro that gives instructions in one window while you experiment in another. See for yourself why no other macro language comes close.

Call for your free evaluation copy today. See why VEDIT PLUS has been the #1 choice of programmers, writers and engineers since 1980.

Supports the IBM PC, XT, AT and PS/2 including DESQview, Microsoft Windows, PC-MOS/386, Concurrent DOS and most networks. Also available for MS-DOS, CP/M-86 and FlexOS. (Yes! We support windows on CRT terminals.) $185.

FREE EVALUATION COPY*
Call 1-800-45-VEDIT

- Fully Network Compatible
- Call for XENIX and OS/2 versions
- 30 Day Money-back guarantee

Features of VEDIT PLUS 3.0

- Simultaneously edit up to 37 files of unlimited size.
- Variable sized windows; multiple windows per file.
- Execute DOS commands and other programs.
- Flexible "cut and paste" with 36 "scratch-pad" buffers.
- Block operations by line, character or column.
- Search with pattern matching or regular expressions.
- Configuration—determine your own keyboard layout, create your own editing functions, support any screen size.
- Select window colors, support 43 line EGA, 50 line VGA.

EASY TO USE
- Modern pull-down menu system. Pop-up ASCII table.
- Context sensitive on-line help is user changeable.
- Multi-level Undo (100 to 1000 levels). Undo keystroke by keystroke or line by line.
- On-line integer calculator (also algebraic expressions).
- Keystroke macros speed editing, menu function "hot keys."

FOR PROGRAMMERS
- Automatic Indent/Undent for "C," PL/I, PASCAL.
- Match/check nested parentheses, e.g. "{" and "}" for "C."
- Flexible macro runs popular compilers and automatically moves cursor to each error in your program. Easily changed to support new compilers and assemblers.

FOR WRITERS
- Word wrap, paragraph formatting and justification.
- Convert to/from Wordstar and mainframe files.
- Flexible printing; fully adjustable margins and Tab stops.

MACRO PROGRAMMING LANGUAGE
- If-then-else, looping, testing, string compare, branching, user prompts, keyboard input, 24 bit algebraic expressions.
- Flexible windowing—forms entry, select size, color, etc.
- Simplifies complex text processing, formatting, conversions.
- Extensive 400 page manual with hundreds of examples.

*Also available for TI Professional, Tandy 2000, DEC Rainbow, WYSE 700, Amdek 1280 and Others.
*Free evaluation disk is fully functional and can even edit small files.

Reader Service Number 7

CompuView

1955 Pauline Blvd., Ann Arbor, MI 48103
(313) 996-1299 • Telex 701821 • Fax (313) 996-1308

QUALITY PRODUCTS AT COMPETITIVE PRICES!

CASES & POWER SUPPLY

150 Watt Power Supply (XT)	50.00
200 Watt Power Supply (AT)	80.00
XT Slide Case	34.00
XT Slide with Lock & LED	38.00
STD AT or 80386 w/Lock & LED	65.00
Tower AT or 80386 w/Lock, LED and Power Supply	239.00

MONITORS

EGA/CGA (Auto Switch, .31 dot)	395.00
VGA/EGA/CGA Multi Sync (.31)	495.00
CGA Color	249.00
Amber 12″ TTL	89.00
Green 12″ TTL	89.00
VGA Analog (Mitsubishi .28 dot)	495.00

VIDEO CARDS

Color/Graphics/Parallel	52.00
Mono/Graphics/Parallel	49.00
EGA, CGA, VGA (640x480)	129.00
VGA Analog, STB Extra	239.00

EXPANSION CARDS

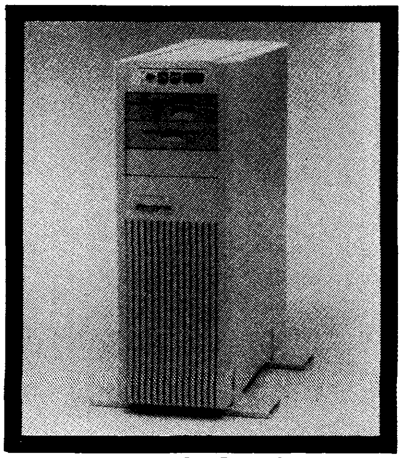
Clock Card	22.00
Dual Floppy Disk Controller	25.00
Game Port	19.00
XT Multi-Function, 1 ser/par/clk/game/2 floppy	61.00
Parallel (Supports LPT 1, 2 or3)	18.00
Dual Serial Port Card — 1 installed switchable Com 1, 2, 3, or 4	22.00
Kit for 2nd Port	20.00
XT 640K RAM (ØK installed)	35.00
XT 2 MB Intel EMS (ØK installed)	99.00
XT/AT Multi I/O Serial/Par/Game	41.00
Kit for Second Serial	30.00
AT 2 MB Intel EMS (ØK installed)	139.00
DTK 4 MB 32 bit memory card w/o RAM	139.00
XT/AT Multi Drive Controller — Supports 1.44, 720K, 1.2, 360K drives on XT or AT	54.00

MOTHERBOARDS

XT/Turbo 4.77/8 mhz	89.00
XT/Turbo 4.77/10mhz	99.00
AT 6/10 mhz Choice of Award, Phoenix or DTK Bios	279.00
AT 6/12 mhz Choice of Award, Phoenix or DTK Bios	340.00
Baby AT 6/12 mhz AMI Bios	299.00
80286 6/16 mhz DTK Bios	$Call
80386 8/16 mhz/DTK Bios	1195.00
80386 8/20 mhz/DTK Bios	1395.00
For XT/AT memory	$Call

FLOPPY DISK DRIVES

Toshiba 360K	89.00
Toshiba 1.2 MB	110.00
Toshiba 3½″ Drive Kit 720K	110.00
Toshiba 3½″ Drive 1.44mb	145.00


— Tower Case Pictured —

KIT OPTIONS

MS DOS 3.21 w/GW Basic	49.00
MS DOS 3.31 w/GW Basic	95.00
*5339 Keyboard Sub	24.00
*Color Options: (Includes video card & monitor)	
CGA Color	200.00
CGA/EGA Color	380.00
CGA/EGA/VGA Color	450.00
ASSEMBLY AND TESTING	
XT Systems	60.00
AT/80386 Systems	80.00

XT KIT W/ 2 Floppy Drives.

Includes: 64ØK RAM, Serial, parallel and game ports, clock/calendar, AT-Style keyboard, cabinet, power supply, mono graphics card and amber or green monitor. Keyboard switchable turbo.

8 mhz with lock, LED, Reset & Turboswitch	825.00
10mhz with lock, LED, Reset & Turboswitch	895.00

XT KIT W/20MB Hard Drive.

Includes: 64ØK RAM, Serial, parallel and game ports, clock/calendar, AT-Style keyboard, cabinet, power supply, mono graphics card and amber or green monitor. Keyboard switchable turbo.

8 mhz with lock, LED, Reset & Turboswitch	1095.00*
10mhz with lock, LED, Reset & Turboswitch	1165.00*

*(For 30MB Miniscribe add $15.00)

80386 KIT —

Includes: 1MB RAM, 1 360K floppy drive, 1-1.2 MB FD, 1 40MB HD, DTK bios, switchable keyboard, monochrome monitor, monographics. Serial/parallel ports, case, power supply, game port, clock/calendar. Main board made in U.S.A.

8/16 mhz	2795.00
8/20 mhz	2995.00

80286 - AT KIT

Includes: 64ØK RAM, 1.2 MB FD, 1 360K floppy drive and 40 MB Miniscribe 3650 HD, 6/10mhz, serial, parallel and game ports, clock/ calendar, AT-style keyboard, cabinet, power supply, monographics card, amber or green monitor, keyboard switchable turbo.

40 MB HD (MFM)	1525.00
60 MB HD (RLL)	1595.00

KEYBOARDS

5339 Professional XT-AT w/12 function key	69.00
5060 Keyboard AT Style	49.00
KB101 Keytronic	67.00
Focus 101 Key, Tactile, Switchable, control caps lock, Dust cover	89.00
—Chosen #1 Find by Micro C Staff—	
101 Key Click Style	59.00

—All keyboards switchable to XT or AT—

HARD DRIVES & CONTROLLERS

AT 40 MB Miniscribe 3650 (61 ms)	375.00
AT 40 MB Miniscribe 3053 (25 ms)	489.00
AT 71 MB Miniscribe 6085 (28 ms)	695.00
AT (MFM) Hard Drive & floppy controller (WD)	130.00
AT RLL HD & FD controller	189.00
XT20 MB Miniscribe 8425 (65 ms)	279.00
with controller	340.00
XT30 MB Miniscribe 8438 (65 ms)	299.00
with controller	355.00

SOFTWARE

MS DOS 3.21 w/GW Basic	49.00
Flight Simulator 3.0	39.95

ACCESSORIES

Generic Analog Joystick	25.00
Gravis Analog Joystick	49.95
V20-8mhz	14.00
1200 Baud Internal	79.00
2400 Baud Internal	119.00
EPROM Programmer (4 gang)	175.00
Memory Chips	**(call for prices)**
MICE	
Logitech 2 Button (serial)	59.00
Logitech 3 Button (serial or bus)	89.00
Logitech HiRez (bus)	109.00
With mouse purchase only —	
First Publisher	50.00
Generic CAD 3.0	50.00
Paintshow	15.00

Prices are subject to change without notice. Shipping CHARGES will be added.

BUILDING YOUR OWN CLONE V2.1
****FREE BOOKLET****
*90-day warranty/30-day money back (subject to restrictions)

Free Instructions with Each System

MicroSphere INC COMPUTERS
P.O. Box 1221
Bend, Oregon 97709
(503) 388-1194
Hours: Monday-Friday 9:00-5:30

MasterCard VISA

Reader Service Number 2

All the speed and power of a hardware-assisted debugger at a software price

Hardware-level break points
REAL-TIME break points on memory locations, memory ranges, execution, I/O ports, hardware and software interrupts. More powerful break points than ANY software-only debugger on the market. Soft-ICE gives you the power of an in-circuit emulator on your desk.

Break out of hung programs
With a keystroke - no external switch necessary. Even with interrupts disabled.

Breaks the 640K barrier
Soft-ICE uses ZERO bytes of memory in the first 1MB of address space. This is especially useful for those subtle bugs that change when the starting address of your code changes. With Soft-ICE your code executes at the same address whether the debugger is loaded or not.

Works with your favorite debugger
Soft-ICE can be used as a stand-alone debugger or it can add its powerful break points to the software debugger you already use. You can continue to use your favorite debugger until you require Soft-ICE. Simply pop up the Soft-ICE window to set powerful real-time break points. When a break point is reached, your debugger will be activated.

Solve tough systems problems too
Soft-ICE is ideal for debugging TSRs, interrupt handlers, self booting programs, DOS loadable device drivers, non-DOS operating systems, and debugging within DOS & BIOS. Soft-ICE is also great for firmware development because Soft-ICE's break points work in ROM.

How Soft-ICE Works
Soft-ICE uses the power of the 80386 to surround your program in a virtual machine. This gives you complete control of the DOS environment, while Soft-ICE runs safely in protected mode. Soft-ICE uses 80386 protected mode features, such as paging, I/O privilege level, and break point registers, to provide real-time hardware-level break points.

"Soft-ICE is a product any MS-DOS developer serious enough to own a 386 machine should have."
Dr. Dobb's Journal —May 1988

Both require 80386 AT compatible or IBM PS/2 Model 80. MagicCV requires at least 384K of extended memory. CodeView is a trademark of Microsoft Corporation.

RUN CODEVIEW IN ONLY 8K!

CodeView is a great integrated debugger, but it uses over 200K of conventional memory. MagicCV uses advanced features of the 80386 microprocessor to load CodeView and symbols in extended memory. This allows MagicCV to run CodeView using less than 8K of conventional memory on your 80386 PC.

Don't let 640K be your limit!
If you are closing in on the 640K limit and would like the power of CodeView, MagicCV is for you.

Don't let the debugger hide the bug!
Even if you're not closing in on the 640K limit, running CodeView with MagicCV makes your debugging environment much closer to the end user's program environment. You can use CodeView to locate subtle bugs that only occur when there is plenty of free memory, or those difficult bugs that only occur when your program is running with a couple of TSRs loaded.

How MagicCV works
MagicCV uses the 80386 to create a separate virtual machine for CodeView. MagicCV uses between 4K & 8K of conventional memory as a bridge between the DOS environment and CodeView.

MagicCV is easy to use
If you are a CodeView user, you already know how to use MagicCV too. Just type MCV instead of CV; everything else is automatic.

MagicCV	$199
Soft-ICE	$386

Buy Both and Save $86!

CALL TODAY
(603) 888 - 2386
***or FAX* (603) 888 - 2465**

30 day money-back guarantee
Visa, Master Card and AmEx accepted

P.O. BOX 7607 • NASHUA, NH 03060-7607

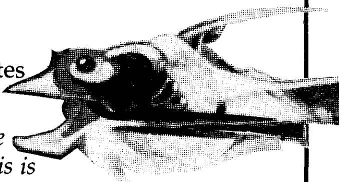
Reader Service Number 110

MagicCV with Soft-ICE
Using Soft-ICE with CodeView gives you the features necessary for professional level systems debugging. MagicCV and Soft-ICE can work in concert with CodeView to provide the most powerful debugging platform you will find anywhere.

THE MICRO TECHNICAL JOURNAL

MICRO CORNUCOPIA

NOVEMBER/DECEMBER 1988 - ISSUE NO. 44



Cover illustration by Rob Sanford

THE MICRO TECHNICAL JOURNAL
MICRO CORNUCOPIA

Editor and Publisher
David J. Thompson

Associate Editors
Gary Entsminger
Cary Gatton

Technical Department
Larry Fogg

Director of Advertising & Distribution
Jackie Ringsage

Accounting
Sandy Thompson

Order Department
Tammy Westfall

Graphic Design
Carol Steffy

MICRO CORNUCOPIA (ISSN 0747-587X) is published bi-monthly for $18 per year by Micro Cornucopia, Inc. 155 NW Hawthorne, Bend, OR 97701. Second-class postage paid at Bend, OR and additional mailing offices. POSTMASTER: Send address changes to MICRO CORNUCOPIA, PO Box 223, Bend, OR 97709.

SUBSCRIPTION RATES:

1 yr. (6 issues)	$18.00
2 yr. (12 issues)	$34.00
3 yr. (18 issues)	$48.00
1 yr. Canada & Mexico	$26.00
1 yr. Other foreign (surface)	$36.00
1 yr. Foreign (airmail)	$50.00

Make all orders payable in U.S. funds on a U.S. bank, please.

CHANGE OF ADDRESS: Please send your old label and new address.

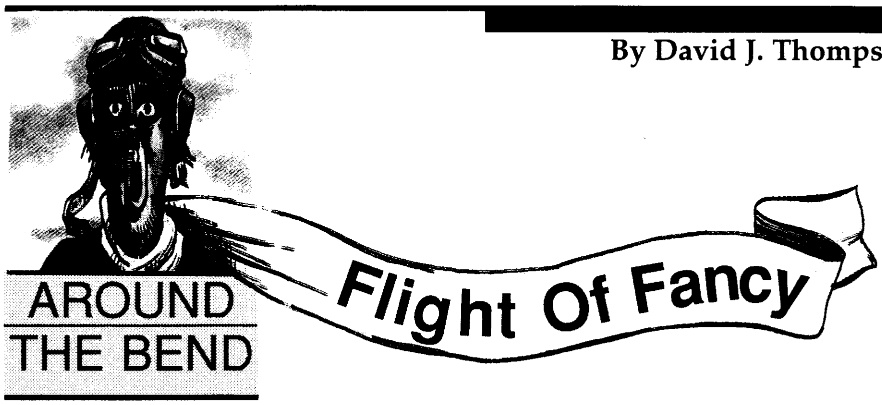
MICRO CORNUCOPIA
P.O. Box 223
Bend, Oregon 97709

CUSTOMER SERVICE: For orders and subscription problems call (503) 382-8048, 9 a.m. to 5 p.m., Pacific time, Monday-Friday.

TECHNICAL HELP: call (503) 382-8048, 9 a.m. to noon Pacific time, Monday-Friday.

RBBS - 24 hrs. 300-1200-2400 baud
8Bits, No Parity, 1 Stop Bit
503-382-7643

ISSN 0747-587X



By David J. Thompson

Sunday 5 a.m., Newburg

I awoke with a start — the tent around me and the earth beneath me were shaking. The roar of a giant piston engine was overwhelming and getting louder. Just as I was certain I would be hacked to shreds, the noise dropped in pitch and receded rapidly. The tent and I were left shaking.

Wide awake, I unzipped the door and tentatively peeked through. It was a cool, grey pre-dawn with clouds draped heavily over the tops of the wooded hills.

I could see the spray plane disappearing through a divide in the trees, wings almost brushing branches, its unmuffled 450 horses still echoing.

Finally: it's quiet.

There are a few quiet coughs from other tents, I'm not the only one awake. Ah, the life of a flying vagabond.

This is it, antique airplanes driven by (mostly) antique pilots. We're sleeping under our wings as we barnstorm Oregon, stopping at everything from informal grass strips to county airparks. (The county airparks feature such luxuries as paved runways and chemical johns.)

Yesterday, the first day of the week-long flyaround, I was part of a formation flying at 1,000 feet, watching cars, watching people, carefully watching the three

Stinson & tent at Newburg, tour's first overnight stop.

Continued on page 69

You can deliver it before 1991?

New Version 2.2
Compatible with DOS 4.0
Uses less memory, runs faster, and adds communications support

It's coming. An era of more powerful PCs. Easier to use PCs. With graphics and character-based programs working side by side. Talking to each other. Multitasking. Windowing. Menuing. Mousing. Letting your customers get their work done easier and faster.

Sell it all now.

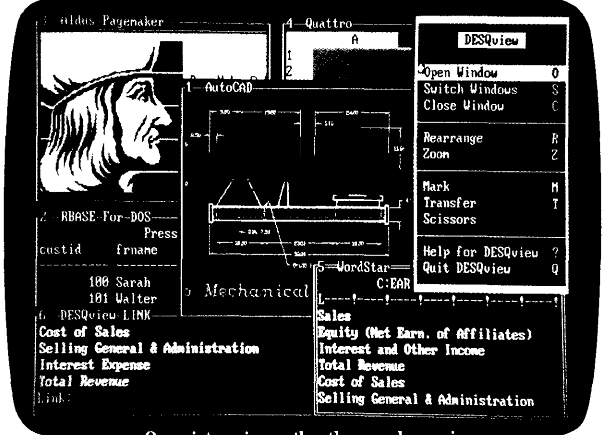
DESQview™ is the operating environment that gives DOS the capabilities of OS/2.™ And it lets your customers, with their trusty 8088, 8086, 80286, or 80386 PCs, leap to the productivity of the next generation. For not much money. And without throwing out their favorite software.

Add DESQview to your customers' PCs and it quickly finds their programs and lists them on menus. So they can just point to the program, using keyboard or mouse, to start it up. DESQview knows where that program lives. And what command loads it.

For those who have trouble remembering DOS commands, it adds menus to DOS. It even lets them sort files and mark specific files to be copied, backed-up, or deleted— all without having to leave the program they're in.

Best of all, DESQview accomplishes all this with a substantial speed advantage over any alternative environment.

For programmers, DESQview's API, with its strengths in inter-task communications and multitasking, brings a quick and easy way to adapt to the future. With the API's mailboxes and shared programs, programmers are able to design programs running on DOS with capabilities like those of OS/2.



```
custid    frname
-------   --------
    100 Sarah
    101 Walter
Cost of Sales
Selling General & Administration
Interest Expense
Total Revenue
Sales
Equity (Net Earn. of Affiliates)
Interest and Other Income
Total Revenue
Cost of Sales
Selling General & Administration
```

One picture is worth a thousand promises.

Multitasking beyond 640K.

When your customers want to use several programs together, they don't have to leave their current program. Just open the next program. View programs in windows or full screen. Open more programs than they have memory for. And multitask them. In 640K. Or if they own a special EMS 4.0 or EEMS memory board, or a 386 PC, DESQview lets them break through the DOS 640K barrier and multitask. For instance, they can start 1-2-3 calculating and tell Paradox to print mailing labels while they're writing a report in Word Perfect, or laying out a newsletter in Ventura Publisher, or designing a building in AutoCAD.

DESQview even allows them to transfer text, numbers, and fields of information between programs.

Fulfill the 386 promise.

For 80836 PC users, DESQview becomes a 386 control program when used in conjunction with Quarterdeck's Expanded Memory Manager (QEMM)-386—giving faster multitasking as well as virtual windowing support.

And when you use DESQview on an IBM PS/2™ Model 50 or 60 with QEMM-50/60 and the IBM Memory Expansion Option, DESQview gives multitasking beyond 640K.

Experts are voting for DESQview. And over a million users, too.

If all of this sounds like promises you've been hearing for future systems, then you can understand why so many VARs and system integrators have chosen DESQview.

PC MAGAZINE EDITOR'S CHOICE NOV. 24, 1987

INFO WORLD 1986 AND 1987 PRODUCT OF THE YEAR

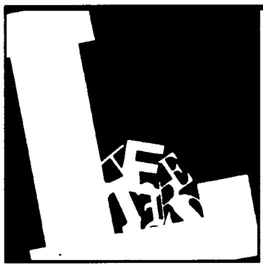
And why *PC Magazine* gave DESQview its Editor's Choice Award for "The Best Alternative to OS/2," why readers of *InfoWorld* twice voted DESQview "Product of the Year" why, by popular vote at Comdex Fall for two years in a row, DESQview was voted "Best PC Environment" in *PC Tech Journal's* Systems Builder Contest.

DESQview lets you sell it all now.

DESQVIEW SYSTEM REQUIREMENTS:

IBM Personal Computer and 100% compatibles (with 8086, 8088, 80286, or 80386 processors) with monochrome or color display; IBM Personal System/2• Memory: 640K recommended; for DESQview itself 0-145K• Expanded Memory (Optional): expanded memory boards compatible with the Intel AboveBoard; enhanced expanded memory boards compatible with the AST RAMpage; EMS 4.0 expanded memory boards•Disk: two diskette drives or one diskette drive and a hard disk•Graphics Card (Optional): Hercules, IBM Color/Graphics (CGA). IBM Enhanced Graphics (EGA), IBM Personal System/2 Advanced Graphics (VGA)• Mouse (Optional): Mouse Systems, Microsoft and compatibles• Modem for Auto-Dialer (Optional): Hayes or compatible• Operating System: PC-DOS 2.0-4.0; MS-DOS 2.0-4.0• Software: Most PC-DOS and MS-DOS application programs; programs specific to Microsoft Windows 1.03-2.1, GEM 1.1-3.0, IBM TopView 1.1• Media: DESQview 2.0 is available on either 5-1/4" or 3-1/2" floppy diskette.

Quarterdeck
150 Pico Blvd., Santa Monica, CA 90405
(213) 392-9851 Fax: (213) 399-3802
Reader Service Number 109



Letters

SOG Thanks

Although this is rather late for a thank you letter, I want you to know that as a computer nerd, idiot, tag-a-long, or whatever appropriate noun would apply, I really enjoyed SOG VII.

First there was the beautiful drive up from Southern California, including Burney Falls, Mt. Shasta, and Crater Lake. Then, the happy surprise of discovering the beauties of Bend. And finally, gaining an appreciation for the wonderful job your crew did in organizing that remarkable get-together.

The Central Oregon Community College provided a beautiful setting for the sessions (even to including an exhibit to grace the art gallery). I was overwhelmed by Debee's demonstration — surely her story should be published. Such an inspiration her accomplishments are!

This is the second letter that I've *ever* attempted on Pat's computer. It's a halting, timid effort but I'll keep trying. Many thanks for an experience I will not forget.

Claire Lewis
8335 Camino Sur
Cucamonga, CA 91730

Editor's note: You're very welcome, Claire. Hope to see you here next year.

Big Al Speaks

I think that I'll remember SOG VII as the "missing" SOG. There were so many things missing this time that I feel it's appropriate.

The sense of excitement that I've felt (and heard others speak of) was missing. It seemed to be a very nondescript gathering. Missing was the understanding of what SOG meant: I overheard a number of attendees ask and wonder what it stood for. (Semi Official Get-Together.)

Absent was the staff vs. readers volleyball game. We all played but the majority of the staff didn't. An SRO hall was waiting for the missing speaker from Intel. Missing was the extra rank of vendor tables in the large hall. Sorely absent were the two plus hours Allyn Franklin needs and deserves for his drive discussion.

Another area of shortfall was the dearth of simple, beginner-oriented topics. If it were my first ever SOG, I might think twice about returning, since the topics were of such an expert level. One last "missing" was the absent moderator for the software forum. It really needed a non-speaker to keep the show and the flow going.

On the plus side, the Bar-B-Q and the lunches were pulled off exceptionally well, another tip of the hat to the COCC staffers. The Micro C forum was also exceptional. I think that it should become a standard topic. I overheard one comment about the forum however, and that was, "Why aren't the gals up there, too?"

Speaking of "the gals," thank you Laura, for all that you did to bring about the SOG. I hope to see you next year! Another thank you to Debee and "Duchess" (I hope I got that right) for showing us that a handicap is really in the eye of the beholder. The rest of the speakers were up to standard SOG excellence, and a hearty thank you to each and every one.

I'd like to offer up a couple of topics for the next SOG to consider. Bruce Eckel should do an RS232C lecture. Someone should do a bit on Viruses and Vaccines, perhaps the fellow who wrote Vaccine. A talk by a SYSOP on dealing with Trojan Horse routines and other nasty uploaded bombs. Some talks on systems other than the MS-DOS group. How about another talk by someone from an outfit that makes computers that make a real difference in people's lives, such as ZYGO from last year.

One last suggestion: since over half of the topics had standing room only even before the lecture began, maybe having three speakers at the same time might help. I understand that there really aren't any larger facilities available, so dividing the listeners up a bit more might give some more room. (Of course the down side is that the speakers love to see a packed room.)

Thanks to all of the staffers and crew that put on the SOG. You all did just fantastic. I'll be back for the next one.

Al (Spike the Ball) Szymanski
8991 Edcliff Ct. SE
Aumsville, OR 97325-9549

Editor's note: Comments noted. The staff was a bit reticent about playing volleyball after seeing you on the opposite side of the net. We're going to be taking Karate-ball lessons this winter, so look out next year.

We already get complaints because there are two talks at a time, three would be about as popular as poison oak at a nudist camp.

You're right about Debee and Christy (and Debee's seeing eye dog). Every SOG has its surprise and they were the surprise this year. A delightful surprise. See the editorial for more on these two.

Allyn Franklin is a problem. I'm not sure whether it's the subject (drive repair) or the person. Probably both. I remember giving Allyn the last afternoon spot last year, that way he could go an extra hour if necessary. I think he finished up the following day. Everyone was still there. We may have to give him the final spot in the show so he can just wrap around to the next year.

As for getting the gals into the Micro C forum, good idea. In fact, they insisted we (Larry, Gary, Bruce and I) couldn't have speaker T-Shirts because four of us were sharing the stage. If they were participating too, I think we'd see a major rule change (they wouldn't miss a chance to get T-Shirts)

Continued on page 78

ERAC CO.

8280 Clairemont Mesa Blvd., Suite 117
San Diego, California 92111
(619) 569-1864

AT

Motherboard 6 & 10 Meg
Zero Wait State
8 Expansion Slots
640K RAM On-Board
Math Co-processor Option
Phoenix Bios
200 Watt Power Supply
Hercules Compat. Video Bd.
Parallel Port
2 Serial Ports Active
Game Port
Clock/Calendar
Hard Disk & Floppy Controller
20M Hard Drive
1.2M 5¼" Floppy Drive
360K 5¼" Floppy Drive
5061 Keyboard
Case with Turbo & Reset,
Hard Drive Light and
Keyboard Disable Switch
Amber Graphics Monitor

$1581

EGA ADD $449
40M HD ADD $150
6 & 12 MHz ADD $73

BABY AT

Motherboard 6 & 10 Meg
Zero Wait State
8 Expansion Slots
80286 Processor
Math Co-Processor Option
640K RAM On-Board
Phoenix Bios
200 Watt Power Supply
Hercules Compat. Video Bd.
Parallel Board
2 Serial Ports Active
Game Port
Clock/Calendar
Hard Disk & Floppy Controller
20M Hard Drive
1.2M 5¼" Floppy Drive
360K 5¼" Floppy Drive
5061 Keyboard
Mini AT Case with Turbo &
Reset, Hard Drive Light and
Keyboard Disable Switch
Amber Graphics Monitor

$1531

EGA ADD $449
40M HD ADD $150

XT/TURBO

Motherboard
5 & 8 MHz Switchable
8088 — V20 Optional
Optional Co-processor
8 Expansion Slots
ERSO or Bison Bios
640K RAM
150 Watt Power Supply
Hercules Compat. Video Bd.
Parallel Board
2 Serial Ports Active
Game Port
Clock/Calendar
Hard Disk and
Floppy Controller
20M 5¼"Hard Drive
2 ea. 360K 5¼" Floppy Drive
AT Style Keyboard
Standard Slide Case
Amber Graphics Monitor

★ ★

$999

EGA ADD $429
40M HD ADD $150
5 & 10 MHz ADD $21

NiCds

AA Cells .6aH	$1.00
12V Pack AA Cells .6aH	6.50
Sub-C Cells 1.5aH	1.50
12V Pack Sub-C	10.00
Double D Cell 2.5V 4aH unused	8.00

GEL CELLS

12V 1.9ah	$6.50
12V 2.5ah	8.50
D Cell 2.5ah	2.00
12V 24ah	35.00

ROBOTICS

12V DC Gear Motor 1"x3", 30RPM	$7.50
5V DC Gear Motor with Tach 1"x2"	7.50
Joystick, 4 switches, 1" knob	5.00
Z80 Controller with 8-Bit A/D	15.00
Brushless 12VDC 3" Fan	7.50
Capacitor, .47farad 10V 1"x1¾"	4.00
Solar Cells .5V .5A, .8"x1.6"	2.50
High Voltage Power Supply Input: 15-30V DC Output: 100V 400V 16KV	6.50

ELGAR UNINTERRUPTIBLE POWER SUPPLIES

200 Watt MODEL SPR201 $129 — 400 Watt MODEL SPR401 $149 THESE SUPPLIES MAY HAVE SOME MINOR COSMETIC DAMAGE, BUT ARE ELECTRICALLY SOUND. SQUAREWAVE OUTPUT. RUN ON INTERNAL OR EXTERNAL 24VDC BATTERY WHEN LINE GOES DOWN. TYPICAL TRANSFER TIME = 12MS.

Battery Supplied — Not Guaranteed **NEW 24V INTERNAL BATTERY $75**

KAYPRO EQUIPMENT

9" Green Monitor	$50.00
Keyboard	75.00
PRO-8 Mod. to your board	149.00
Host Interface Board	15.00
Kaypro II	250.00
KAYPRO IC'S	
81-189 Video Pal	$15.00
81-194 RAM Pal	15.00
81-Series Character Gen. ROMs	10.00
81-Series Monitor ROMs	10.00

POWER SUPPLIES

0-8VDC 100A Metered Volt & Current Regulated	$249.00
5V/1A. -5V/.2A, 12V/1A, -12V/.2A, -24V/.05A	9.90

HOURS: Mon. - Fri. 9 - 6 — Sat. 10 - 4
MINIMUM ORDER — $15.00
TERMS: VISA, MasterCard, Certified Checks, Money Order, NO COD. Visa and MasterCard add3%. *Personal checks must clear BEFORE we ship.* Include shipping charges. California residents add 6½% Sales Tax. For more information please call.

CPU & SUPPORT CHIPS

MC68000-8 CPU	$8.00
Z80 CPU	.75
Z80A CPU	1.50
Z80 CTC	1.50
Z80A PIO	2.00
Z80A SIO	5.00
8088	6.50
8089-3	6.50
D8284A	2.50
SIP DRAM 256-12	7.00
4164-10	2.50
4164-12	2.10
1793	6.00
1797	7.00
6845	5.00
VC3524 Switching Regulators	5.00
1458 Dual Op-AMP	.70
LM2877P 4W Stereo Amp Dual	2.50
MB81464-15	2.75
2716	3.00
2732	3.25
2764	3.50
27C128-1	9.00
74HC00	.38
74LS125	.30
74LS373	.50
74LS174	.30

SWITCHERS

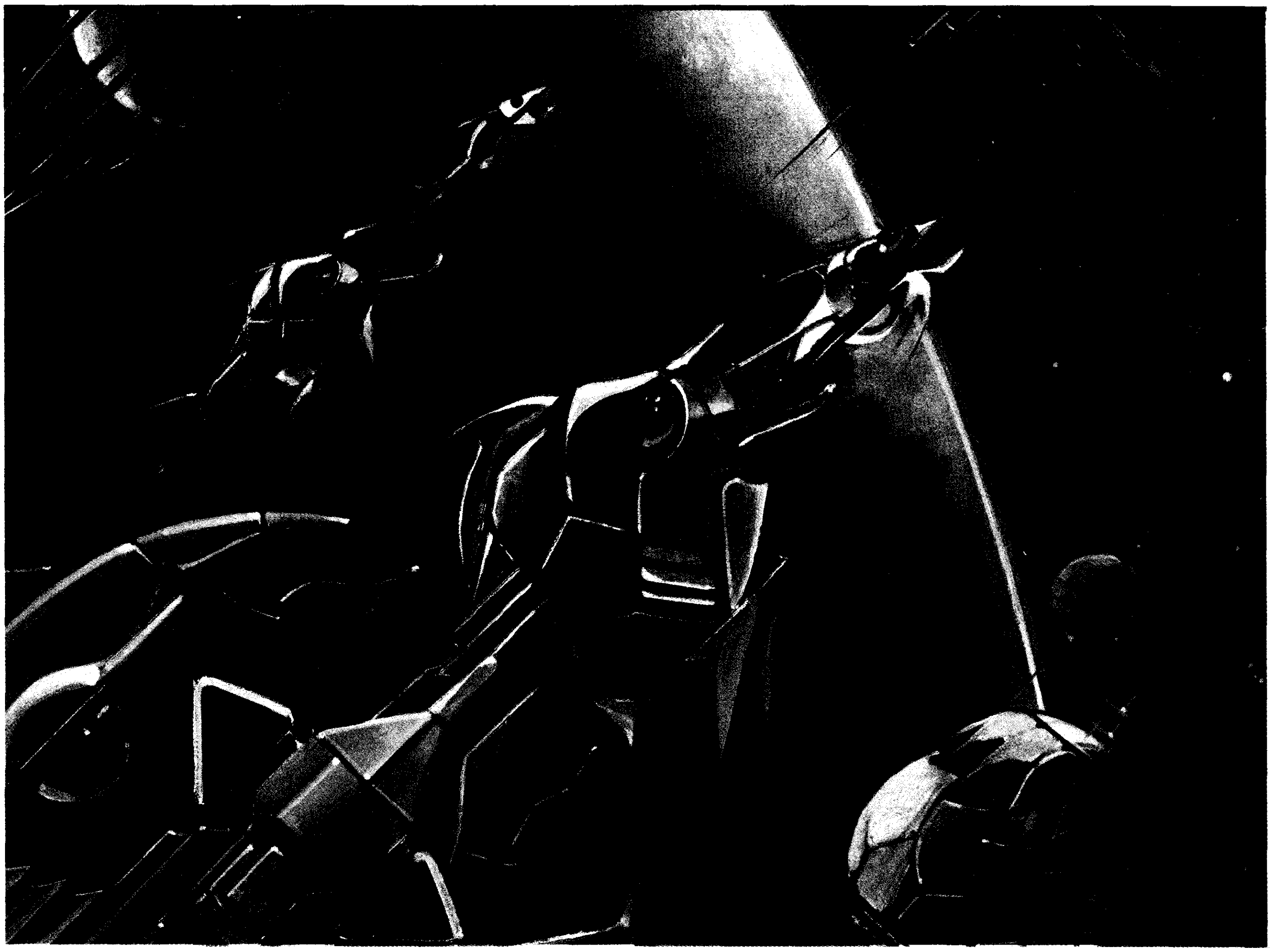
5V/9.5A, 12V/3.8A, -12V/.8A	$39.00
5V/3A, 12V/2A, -12V/.4A	19.50
5V/6A, 12V/2A, -12V/1A	29.00
5V/6A, 24V/1¼A, 12V/.6A, -12V/.6A	29.00
5V/10A	19.00
5V/20A	24.00
5V/30A	39.00
5V/75A, 12V/8A, 24V/5A	55.00
5V 100A	100.00
5V 120A	110.00

TEST EQUIPMENT

OSCILLOSCOPES	
TEK 7403N/7A18N/7B50A 60 MHz	$650
ANALYZERS	
TEK 491 10MHz - 40 GHz	$4500.00
Nicolet 500A 1 Hz 0-100 KHz	1800
Biomation 805 Waveform Rcrdr	259.00
Biomation 8100 2-Channel Waveform Recorder	795.00
HP1600A Logic Analyzer	600.00
HP1600A/1607A Logic Anlyzr	1000.00

DBASE BOOK OF BUSINESS APPLICATIONS *by Michael J. Clifford*
Reg. $19.95 ***NOW ONLY*** **$3.00**

Reader Service Number 93



A Taste Of Smalltalk:

Object-Oriented Programming From The Bottom Up

One of the things that surprised me as I read through this and the other object-oriented articles in this issue, was the similarities among languages. The ideas of objects, classes, and messages are definitely key. It's where things are bound (fancy term for when space is allocated) that seems to change from interpreter (Smalltalk) to compiler (C++). Check out all these pieces, I think you'll really get a sense of these unique languages.

In the early 70s, the Xerox Palo Alto Research Center in California began developing a new programming language — Smalltalk. Several versions later, in 1981, they released Smalltalk-80. In August '81, *BYTE* devoted an entire issue to Smalltalk-80, introducing the micro computer world to object-oriented programming.

There are now several versions of Smalltalk; we list them (and other sources) at the end of this article.

What's In A Smalltalk?

When you buy a Smalltalk, you get:

By Jan Steinman and Barbara Yates
Tektronix, Inc.
P.O. Box 500, Mail Sta. 50-470
Beaverton, OR 97077

(1) A virtual image file which contains the source code for a group of core classes.

(2) A starter set of programming tools.

(3) An interpreter.

You can also add on packages of additional classes. Classes are the voices of Smalltalk.

Smalltalk is both a programming language and a programming environment, consisting of, at least: an editor, a debugger, and an interface to a file system. Ask any Smalltalk programmer what's special about Smalltalk, and he'll invariably mention how much fun it is.

Fundamental Talk

Basically, we program in Smalltalk by sending "messages" to "objects," which are "instances" of "classes."

Some classes represent fundamental objects such as integers, characters, and reals (or floating point numbers). The names of classes are always capitalized. So the corresponding classes for these basic types are Integer, Character, and Float.

Other classes represent more abstract objects. For example, Process and Stream. There's also a rich set of classes of Collections: for instance, Array, String, and Dictionary.

Your Smalltalk system *is* effectively its classes — from Cursor, to InputSensor, to Compiler, to View. You send messages to instances of classes, and things happen.

There's some confusion (isn't there always?) about classes because there's no standard class library. The same class may go by a different name in a different implementation. And Smalltalk implementations may contain fewer than one hundred or as many as several hundred classes.

Later we'll go a little deeper into classes and clear up some of the mystery.

In Smalltalk, there's no program per se. At least in the beginning, think in terms of modifying the behavior of existing classes, rather than creating new data structures and operations.

Messages

Messages are represented symbolically by selectors. Different types of message selectors correspond to the number of objects involved in the message.

A unary message has one object, the object receiving the message. A binary message has two objects — the receiver and one argument.

Keyword messages are sent to the receiver with one or more arguments. Each part of a keyword selector ends with a colon. For example, when an object can't reply to a message, it sends itself the message —

```
doesNotUnderstand:
```

Precedence

Smalltalk uses precedence rules to resolve ambiguities —

- 1. expressions in parentheses,
- 2. unary messages,
- 3. binary messages,
- 4. keyword messages,
- 5. equal precedence messages are sent left-to-right,
- 6. variable assignment happens last. The symbols ":=" or "<-" (left-arrow) are assignment operators on different systems.

So in the Smalltalk statement —

```
stream := ReadStream on:
 (Array with: 'Joe','Blow') first.
```

the parenthetical expression must be manipulated first.

Within it is a binary message. So the concatenation selector "," is sent to the String 'Joe' along with the String argument 'Blow', resulting in the new String 'JoeBlow'.

This new String is then sent as an argument of the keyword message with: to the class Array, which responds with a new Array object containing the String 'JoeBlow'.

The unary selector "first" is then sent to the newly created Array, returning the first (and only) object in the Array, which is the String, 'JoeBlow'.

This String is then sent as an argument to the on: message to the class ReadStream. ReadStream responds with a new ReadStream object, which is assigned to the variable "stream".

Creating Classes & Behaviors

When we create a class, we (programmers) decide what kinds of data are needed to describe the instances of that class. The private data of instances is stored in instance variables.

All instances of a class can share data, as well. Shared data is stored in class variables.

The programmer decides the behavior of a class, and chooses the selector names of messages to send to the class (or to instances of the class) in order to start the behavior.

Methods are the procedures (or functions) of Smalltalk which execute when a message is delivered. In Smalltalk-80, methods are grouped into categories according to their function.

Programs & Objects

In traditional, structured languages, you write programs by defining data structures and the operations you want to perform on that data.

In Smalltalk, there's no program per se. At least in the beginning, think in terms of modifying the behavior of existing classes, rather than creating new data structures and operations.

An object is a private, persistent state that behaves in a particular manner through a publicly accessible interface.

The arrangement of the state of one object is unknown to other objects. In other words, you program by focusing on the way an object behaves (or manipulates data).

Messages Determine Behavior

The behavior of an object is determined by the messages an object can reply to. You can simulate message passing in many languages, but Smalltalk is the only popular language where message passing is the only means of communication among objects.

It's important to note that a message selector exists apart from an object. So the particular object a message is sent to is very likely indeterminable at the time the method is written.

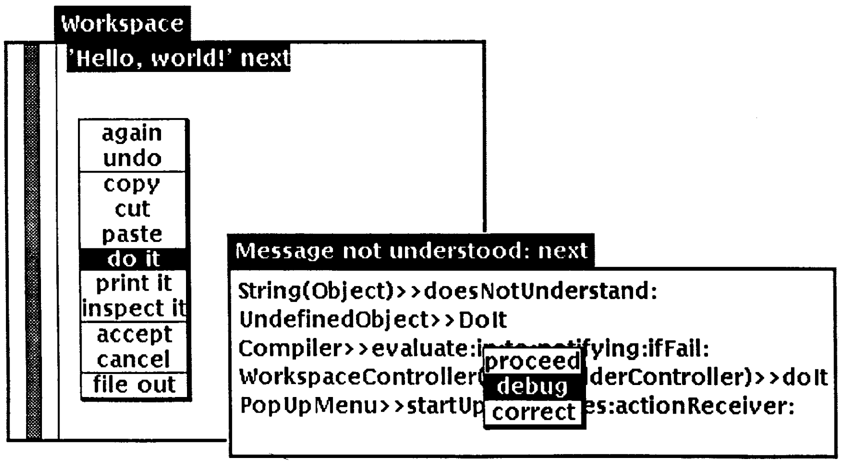
Editor's note: For instance, in C++, memory for an object has to be allocated at compile time, whereas the size of a smalltalk object changes while the program is running.

This polymorphism is the biggest reason there are no Smalltalk compilers in common use, and why those which do exist (in research labs) place restrictions on message passing. It's also cited by backers of late-binding languages (like Smalltalk) as the key that turns a language truly object-oriented. In this sense, neither C++ nor Ada are truly object-oriented.

A Smalltalk message, like printString, can be sent to any object (regardless of class), and the object will respond by identifying itself with a String of some sort.

Modems talk to each other this way. And if you unintentionally reach a modem, when you're trying to reach a fellow human being, the modem will let you know that it doesn't understand by hanging up!

Figure 1 — Smalltalk Run-time Error Notifier.



```
'Hello, world!' next
```

Message not understood: next

```
String(Object)>>doesNotUnderstand:
UndefinedObject>>DoIt
Compiler>>evaluate:i          fying:ifFail:
WorkspaceController(          derController)>>doIt
PopUpMenu>>startUp            s:actionReceiver:
```

In Smalltalk, when an object can't reply to a message, it hangs up, sending itself the doesNotUnderstand: message, which we can intercept for error recovery.

For example, sending the message "next" to the string 'Hello, world' brings up a notifier window. (See Figure 1.)

Inheritance

How does an object know about messages like printString and doesNotUnderstand:? Through inheritance, a key element of any object-oriented language.

The Smalltalk interpreter directly supports hierarchical inheritance.

Like everything else in Smalltalk, classes are objects. You create a new class by sending a class creation message to some other class. Your class becomes a subclass of that other class, inheriting all its behavior.

At the root of the class hierarchy is the class Object, which understands the messages common to all objects, such as printString and doesNotUnderstand:.

And Where Do Objects Come From?

Most real-world objects have fairly long lifetimes. With the exception of radioactive Polonium-214 (which has a half-life of 164 microseconds), or chocolate ice cream (which in our house, has a full life about as long), real objects tend to stick around. Not so with Smalltalk objects.

They may be created hundreds or thousands of times each second. When they're no longer needed, they become garbage. Fortunately, a garbage collector knows when garbage is garbage and reclaims the memory.

In a world of routine object creation and destruction, garbage collection is sorely needed.

So you (the programmer) create objects deliberately. And the Smalltalk system creates objects behind your back, as it answers messages.

While the usual activity of Smalltalk programming is defining and refining the behavior of a class of objects, the usual activity of executing Smalltalk code is creating objects and sending them messages. It's not unusual for a large Smalltalk application to generate hundreds of thousands of objects.

For example, you might do a "print it" on the statement "newString := 'Hello, ', 'world!'". (See Figure 2.)

This sends the String 'Hello,' the concatenation message "," with the String argument 'world!'. It replies (in the best of Smalltalk worlds) with the brand new String object 'Hello, world!', which is stored in the temporary variable newString.

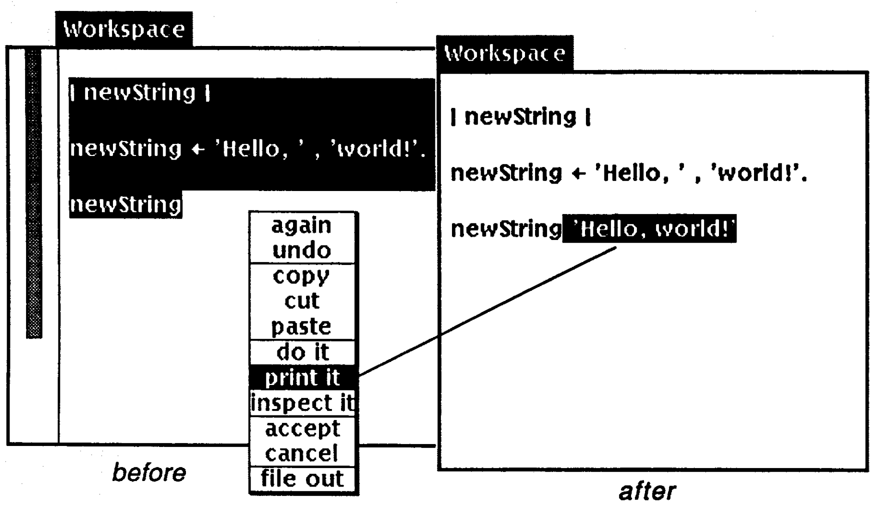
Then, the workspace window closes, and the new String object is collected as garbage.

The One-Line Database

Let's create some more useful objects. (See Figure 3.)

Execute

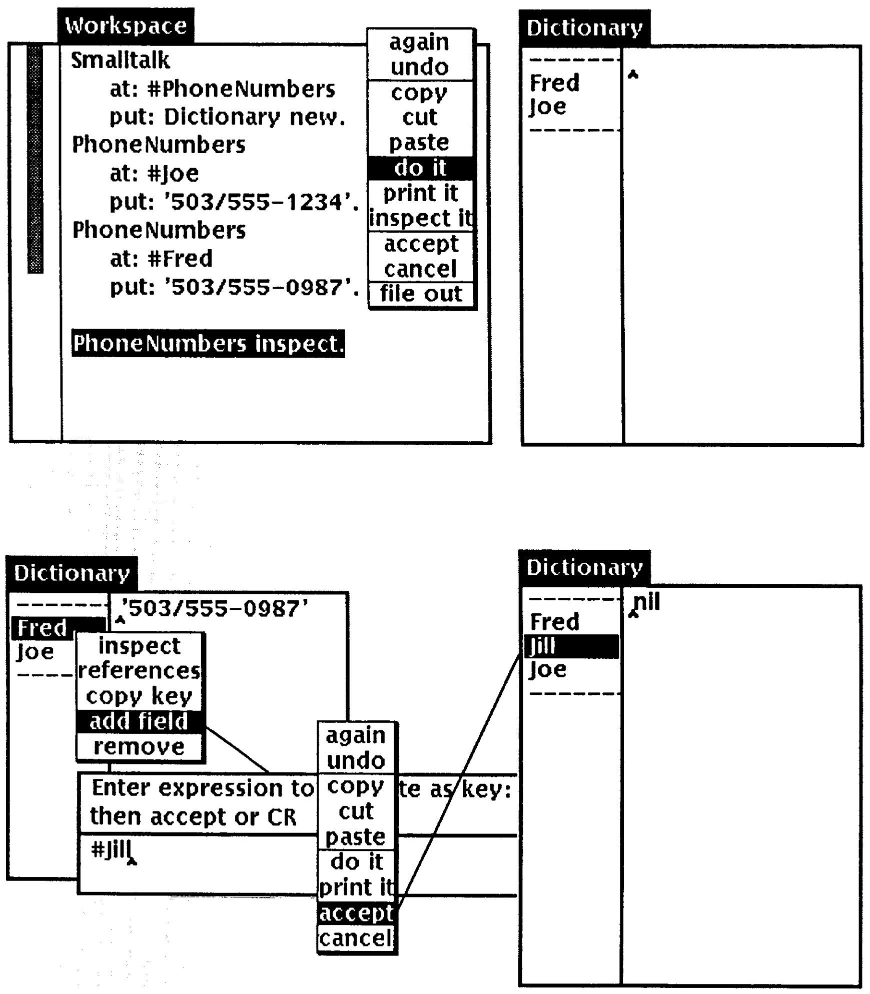
Figure 2 — Create a New Object by Sending a New Message



```
| newString |

newString ← 'Hello, ' , 'world!'.

newString
```

```
| newString |

newString ← 'Hello, ' , 'world!'.

newString 'Hello, world!'
```

Figure 3 — The One-line Database in Action.



```
Smalltalk
    at: #PhoneNumbers
    put: Dictionary new.
PhoneNumbers
    at: #Joe
    put: '503/555-1234'.
PhoneNumbers
    at: #Fred
    put: '503/555-0987'.

PhoneNumbers inspect.
```

```
'503/555-0987'
Enter expression to ... te as key:
then accept or CR
#Jill
```

Eco-C88 C Compiler with Cmore Debugger

Professionals prefer the Eco-C88 C compiler for ease of use and its powerful debugging features. Our "picky flag" gives you nine levels of lint-like error checking and makes debugging easy:

"I'm very impressed with the compiler, editor, and debugger. I've tried quite a few different compilers for the PC and have given up on all of the others in favor of yours . . . I've gotten to the point where I download C code from a DEC VAX/VMS system just to be able to compile it with the picky flag set at 9. It finds lots of things VMS totally ignores . . ."
JS, Oak Ridge, TN

The Eco-C88 compiler includes:
- A full-featured C compiler with 4 memory models (up to 1 meg of code and data) plus most ANSI enhancements.
- Without a doubt, the best error checking you can get. We catch bugs the others miss, making you much more productive.
- Cmore is a full-featured source code debugger, not some stripped-down version.
- Robust standard library with over 230 useful (no "fluff") functions, many of which are System V and ANSI compatible. Full source is available for only $25.00 at time of order.
- CED, a fast, full screen, multiple-window program editor with on-line function help. You can compile, edit, and link from within CED.
- cc and mini-make utilities included that simplifies the most complex compiles.
- Users manual with over 150 program examples (not fragments) to illustrate how to use the library functions.
- Fast compiles producing fast code.

Our Guarantee: Try the Eco-C88 compiler for $99.95. Use it for 30 days and if you are not completely satisfied, simply return it for a full refund. We are confident that once you've tried Eco-C88, you'll never use anything else. Call or write today!

Orders: **1-800-952-0472**
Info: **1-317-255-6476**

Ecosoft Inc.
6413 N. College Avenue
Indianapolis, IN 46220

ECOSOFT

Reader Service Number 9

```
Smalltalk
  at: #PhoneNumbers
  put: Dictionary new.
```

by highlighting the text and choosing "do it" from the menu.

This is Smalltalk for — create a new global variable named PhoneNumbers and place a newly created Dictionary object in it.

Now you can begin adding objects to your new Dictionary.

To retrieve an object from the PhoneNumbers Dictionary, "print it" using the following statement :

```
PhoneNumbers at: #Joe.
```

What's Going On Here?

We're creating objects by sending messages to other objects. The names #Joe and #Fred are actually objects of class Symbol (similar to String), whose instances are unique in the system.

Once you've created #Joe, any mention of #Joe, is a reference to the existing Symbol, #Joe.

When you send the message at:put: to the new Dictionary object PhoneNumbers with two arguments, it creates a new Association object that stores the first Symbol argument together with the second String argument. These key-value pairs are very useful for symbolically representing data.

Although we're using a Symbol as the key and a String as the value in this example, any object can become associated with any other object.

All objects are equipped to answer common messages (in addition to printString and doesNotUnderstand:). One particularly valuable one is "inspect," which forces an object to open a window on its private state. This is so useful during debugging that the standard Smalltalk debugger includes an Inspector for the object whose method is being debugged.

Late-binding of message passing lets all users of an object benefit from modifications.

Executing "PhoneNumbers inspect" will pop up an Inspector that allows you to add, remove, and modify entries in your database. So —

```
(Smalltalk at: #PhoneNumbers put:
  Dictionary new) inspect.
```

creates a simple database manager, complete with user interface! Not bad for a line of code!

An Environment Of Reusable Parts

The key to Smalltalk productivity is that components are reusable.

The Class Inspector is a good example. Programming often requires examining or modifying an object's private state and this is the task of Inspector. (Without modification it's also a debugger.)

"So," you might counter, "my Speedo QuickWindows Mark III Library from Ne'erdowell Software lets me do that in C."

INDUSTRIAL STRENGTH OOPS.

You have three options in today's world; lead, follow or get out of the way. You've already taken a leadership position in hardware with the latest 286 or 386 system. Now you can use that triple-digit architecture to blast ahead of the pack with the most powerful new Object Oriented Programming (OOPS) software on the market: Smalltalk/V286.

Smalltalk/V, the original OOPS tool for the PC, gave scientists, engineers, programmers and educators a brand new way to solve problems. And soon they were developing exciting new applications in everything from economics to medicine to space.

Now Smalltalk/V286 gives you true work station performance with industrial strength capabilities like: push-button debugging; multi-processing; portability between DOS, OS/2 and Presentation Manager operating environments; integrated color graphics; a rich class library; and access to 16 MB of protected mode memory, even under DOS.

The new Smalltalk/V286, which is even easier to learn and use than Smalltalk/V, retails for just $199.95. Or you can buy Smalltalk/V, still the world's best selling OOPS, for only $99.95. And both come with our 60 day money-back guarantee.

Check out the new Smalltalk/V286 at your dealer. If he doesn't have it, order toll free, 1-800-922-8255. Or write to: Digitalk, Inc., 9841 Airport Blvd., Los Angeles, CA 90045. And let us put you ahead of the power curve.

Smalltalk/V286

Reader Service Number 111

Perhaps, but does it let you easily modify and extend the library so that already existing applications can use the changes? Probably not, unless your C is late-binding. Late-binding of message passing lets all users of an object benefit from modifications

To get an idea of how powerful this is, I'll briefly describe the abilities of the major classes in Smalltalk.

Magnitude's Classes

Numeric classes are all subclasses of Magnitude. Magnitude does very little, it's really an abstraction for all classes of objects that know how to compare themselves with each other. Abstract superclasses, like Magnitude, are common in Smalltalk.

Below Magnitude in the hierarchy are the classes that deal with quantities, like Date, Time, Character, and Number. Date and Time contain methods for printing and comparing dates and times, as well as special methods for obtaining partial information, such as day of the week.

The class methods "Date today" and "Time now" return a new object representing the date (or time) a message was sent.

Class Character contains methods for converting a character to or from ASCII, or for determining if it's an alphanumeric, vowel, or whatever. (This class has been successfully modified to support Oriental character sets, with more than 256 characters.)

Class Association, mentioned earlier, might seem out of place as a subclass of Magnitude. But associations connect or associate a symbolic name with an object. And such symbolic names need order. So two Associations need to know how to compare themselves.

Think of Associations as entries in the index of a book, where keywords serve as references to page numbers.

Numbers

Class Number is an abstract superclass for objects which know about arithmetic. You would never create an instance of Number (since it's an abstract class), but through inheritance, Number provides the arithmetic ability you would associate with a number.

Number has subclasses that do most of the work. Float provides floating-point capability, and Fraction represents a ratio between two Numbers. Fractions may seem quaintly archaic. They are, until you have to deal with non-integer quantities that don't lend themselves to floating point.

For example, using Fractions, you can add one penny to the sum of the National Debt. Attempting to do the same using floating point math would result in no change, which is convenient to politicians, but hardly accurate. In particular, Fraction provides a good superclass for creating a fixed-point class.

The Number subclass Integer is another abstract class, which serves as a superclass for LargePositiveInteger, LargeNegativeInteger, and SmallInteger.

For most purposes, it's safe to treat all these the same (with the understanding that larger integers take more memory and execution time).

Most modern systems allow SmallIntegers between -2^{30} and 2^{30-1}, although some older versions of Smalltalk limit SmallIntegers to the range of -2^{14} to 2^{14-1}

Numeric Conversions

One of the nicer things about

Smalltalk/V 286
Object-Oriented Programming System (OOPS)
digitalk inc
World's #1 Best Seller.
More people solve more problems with Smalltalk/V than any other OOPS.

Smalltalk is the way conversions between the numeric classes are handled. With the help of the large integer classes, Smalltalk supports infinite precision, and conversions always attempt to maintain the greatest possible accuracy.

For this reason, executing the simple expression "1/3" returns the Fraction 1/3, which is usually not what you want. Also, if you don't need infinite precision, you'll get better performance by converting intermediate results to Float.

Smalltalk is popular in numerical and business computing, despite its (only partially deserved) reputation for being slow.

Collection's Classes

The next major group of classes are Collection and its subclasses. All Collection subclasses represent objects made up of groups of other objects. Their behavior includes methods for detecting the inclusion or count of a particular object, adding, removing, converting Collections, and enumerating.

Enumeration is very handy. Instead of writing clumsy, hard-to-maintain do-while or for loops, you simply say "execute this block of code for each object in the Collection." It doesn't matter which Collection, or what's in it.

Enumeration methods include do:, which simply executes the Block for each object in the Collection; collect:, which does the same, but saves the result of each execution in a new Collection of the same class as the original; inject:into:, which can accumulate a running value obtained from executing the Block; and select: and reject:, which return a Collection of the same class as the original that contains only those objects for which the Block returned true or false, respectively.

We won't cover all the Collection subclasses, but instead stick to the high points.

Traditional Arrays

If you're more comfortable with the traditional languages, one class in this group will be familiar: Array. Arrays have a fixed size, and their component objects are accessed by an Integer (as in other languages) beginning with the index 1. It would be easy to implement an Array subclass VariableOriginArray, which would allow arbitrary Integers for the starting and ending index, or WrappingArray, which would map illegal indices to legal array positions.

Array subclasses include ByteArray and String, whose elements are eight-bit quantities. ByteArrays are useful for interfacing to hardware or accessing binary data files.

Strings are Arrays of Characters and contain all the methods needed for manipulating text, including comparing, substring testing, searching and replacing, converting, abbreviation, encoding, and displaying.

Although there are many methods in String, strings are by their nature ubiquitous, and the power of Smalltalk is quickly and easily put to use by the beginner by adding useful behavior to Strings.

Ordered Collections absolve the programmer of all responsibility for bounds checking. By increasing the size of an array (etc.) it can accept more objects.

Is There Order?

The Collection subclass OrderedCollection significantly alters the way a C, Pascal, or Fortran programmer must think. OrderedCollections absolve the programmer of all responsibility for bounds checking, by increasing the size of an array (etc.) so it can accept more objects. This is a difficult thing for hybrid object-oriented languages, like C++, Modula-2, or Ada, to accomplish, since they typically don't have garbage collection.

Added behavior includes methods for adding, removing, and accessing objects at the beginning or end of the Collection, or before or after a particular item in the Collection. We have all had the experience of writing and rewriting functions to do these sorts of things in C or Pascal!

OrderedCollection has an extremely useful subclass called SortedCollection, which maintains a collating sequence in its component objects, even through additions and removals. Any comparison algorithm can be supplied to the SortedCollection. Or you can use default algorithm "greater-than-or-equal-to." Generally, the type of objects you store in a SortedCollection will know how to compare themselves to each other, so there's little need to provide special comparison algorithms.

Since SortedCollections are maintained in sorted order, they're hard on performance. But you can easily change their behavior to sort only when asked for their entire contents.

Sets

The last major group of Collections is the Set hierarchy. Unlike most other Collections, Sets (and subclasses) are accessed by a hash probe, rather than by index, and occupation of the Set by its component objects is sparse.

This is the classical space-speed tradeoff — adding objects can be much faster when lots of space is available. Because access is by hash, objects can't be duplicated and their order in the Set isn't maintained. But access is much faster than if a search had to be performed, as in other Collections.

A Dictionary (e.g., a database management system) is a Set of Associations.

You can store any type of object: including multiple-field objects or even other Dictionaries. Dictionary is one of the most useful Collections, and is a good one for beginners to subclass.

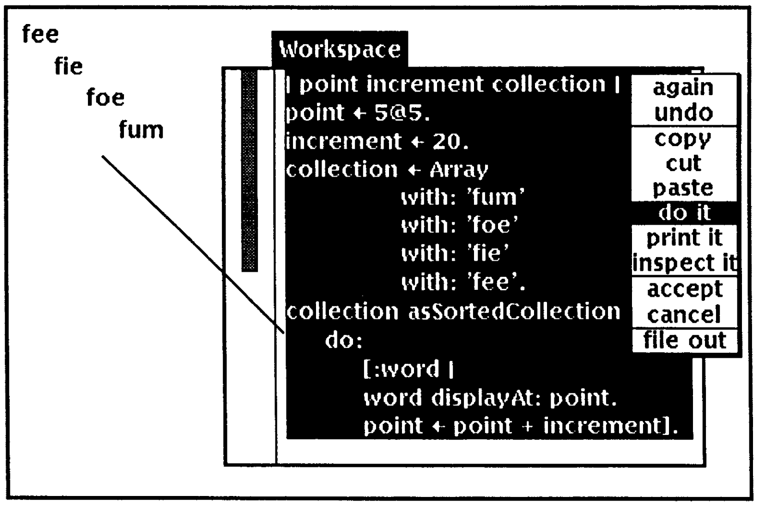
The system uses Dictionaries to associate message names with the code of a method, to store global variables, and to manage system resources. Figure 4 shows a few of the features of Collections at work.

We begin by declaring three temporary variables, point, increment, and collection, and assigning to them a new Point object, the SmallInteger 20, and a new Array object containing four Strings.

Next, we ask the Array to convert itself to a SortedCollection, using the default collating sequence. The resulting SortedCollection is sent the enumeration message do:, with the Block of code within the square brackets as an argument.

The Block is repeated for each object in "collection," and each time it's repeated, the Block variable "word"

Figure 4 — A Few Features of Collections at Work.



```
fee
  fie
    foe
      fum
```

```
| point increment collection |
point ← 5@5.
increment ← 20.
collection ← Array
          with: 'fum'
          with: 'foe'
          with: 'fie'
          with: 'fee'.
collection asSortedCollection
   do:
      [:word |
      word displayAt: point.
      point ← point + increment].
```

takes on the value of the next object in "collection." So, each object in "collection" is displayed on the screen at the location "point," and "point" is incremented by "increment."

Miscellaneous Classes

Before we move up to the classes that do windows and graphics, we need to mention a few housekeeping classes. Stream and its subclasses are invaluable for sequential access to a Collection.

Streams

While all Collections understand enumeration behavior, sometimes it's useful to be able to interrupt, skip forward or backward, or terminate the enumeration of a Collection.

Streams maintain a position in their Collection and are useful for modeling sequential things like files or terminals. Stream, itself, is an abstract class and only provides instance creation. It tests to see if you're at the end of the Collection, does basic enumeration, and provides accessing methods for its subclasses.

ReadStream gives you access to a Collection. The "next" message, for instance, returns the next object in the Collection, and the "next:" message takes an Integer argument and returns the specified number of sequential objects in the Collection.

WriteStream adds write access using the selectors nextPut: and nextPutAll:. ReadWriteStream and FileStream understand both reading and writing messages, one on a Collection, and the other on a file.

Boolean is an abstract superclass, with the subclasses True and False. These objects understand boolean logic. They are the objects that conditional branching and looping messages are sent to.

UndefinedObject has a single instance, known as nil. nil is used to initialize objects. If you get an error notifier that mentions UndefinedObject, chances are you forgot to initialize a variable. And so a message was sent to nil.

Blocks

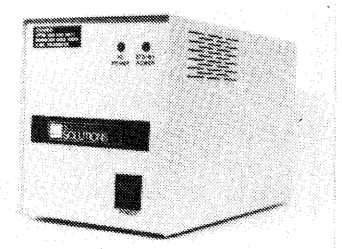
BlockContext (or Context on some systems) is simply a way of moving code around the system. Conditional branching and looping, enumeration, error recovery, and many other functions are created from Blocks. For example, SortedCollections contains a Block of code that compares two component objects in the SortedCollection to determine their order.

Blocks are very useful, but they can also waste a lot of memory and execution time. Blocks can take arguments, which change their state (e.g., during enumeration). The number of arguments specified when the block is created and number sent during execution must be the same.

Graphics

A whole group of classes represent graphic objects. Point, mentioned earlier, represents a position in an area. Rectangle contains two Points marking opposite corners. Form contains a bitmap, making up the graphic image.

UNINTERRUPTABLE POWER SOURCE



MICRO
SOLUTIONS protects your equipment and your data from **power outages** and **brownouts.** Our power systems provide the **fastest switching speed** in the industry (2 ms± 1).

EMI/RFI filtering and surge/spike protection all in one affordable unit. 1 year warranty on all units. Available in a size to suit your needs.

200 watts	$290.00
350 watts	$360.00
550 watts	$410.00
800 watts	$710.00
1000 watts	$810.00

Includes shipping to your door in the continental U.S.. As specialists in overseas systems, we can supply 220 volt units. Call or write for details.

SOFTWARE SPECIAL

ACT!

See what qualifies as GREAT SOFTWARE! **ACT!** does for people what **1 2 3** does for numbers. **ACT!** helps you manage your contacts with the important people in your life.

Call today for more information - or send a $10 (refundable) deposit to receive a copy of the **ACT!** video demo tape. Your deposit will be refunded - whether you purchase **ACT!** or not - when the video is returned in good condition.

RON ANGERT'S
MICRO
SOLUTIONS

WE SHIP WORLDWIDE

C.O.D. MasterCard VISA

Dealers Supported
Drawer B Riner, VA 24149
1-800-323-4829 (703) 382-6624
Call 24 hours - 7 days a week

Reader Service Number 24

BitBlt manipulates Forms, and is how graphics get displayed. Don't use BitBlt as a model for your own classes — it's better used as an example of how Smalltalk code can be non-object-oriented!

User Interface Classes

The most important user interface class is ParagraphEditor on Smalltalk-80 systems and TextEditor on Smalltalk/V systems. It handles text editing and entry for the whole system.

SelectionInListController (ListSelector on Smalltalk/V) provides a simple way to choose an item out of a list. Unlike pop-up or pull-down menus, these lists are scrollable. You'll see these throughout the system as "subviews," or windows within other windows.

The Inspector class (mentioned earlier) is one of the most versatile performers in the user interface group. It allows you to examine and modify the private state of an object. It combines a SelectionInListView, to allow selecting a variable from a list of all variables of the object, with a ParagraphEditor, used to view or modify the selected variable's contents. Its subclass DictionaryInspector allows selection of the inspected object by key.

FileList lets you examine files and directories, and typically includes a list of files or directories and a window for editing a selected file.

The important aspect of having the source code to all the classes that make up the environment is that you can customize it, even the user interface, to suit yourself.

Don't like the way a menu is set up or want more items on it? Change it! Want to change the way the window panes look? Do it! Want to make the interface to the file system work differently? Go ahead!

Decision Points For Potential Users

Primarily, Smalltalk is a language and environment for rapid program development. In other words, it's ideal for hackers. Some studies show that Smalltalk programmers are about five times more productive than Pascal or C programmers, and up to 25 times more productive than folks working in assembly.

On the flip side, Smalltalk applications aren't easy to separate from the environment, and this makes delivering applications difficult. The Smalltalk environment is large, and has a (partially deserved) reputation for being slow.

Smalltalk is also attacked for being wordy, but usually by those who think you need to write a lot of code. Code reusability, combined with copy/cut/paste editing, means that an experienced Smalltalk programmer generally spends less time typing than a C programmer.

"Make it work, then make it fast," is something we all need to repeat to ourselves from time to time. Smalltalk makes it work like no other, and when it comes time to make it fast, *there is help.* All popular versions of Smalltalk include some means for access to assembly code, and Tektronix and Digitalk Smalltalk include full access to all system calls.

If you decide to buy a Smalltalk implementation and work for someone with deep pockets, consider getting a Smalltalk-80 variant. ParcPlace Smalltalk-80 is available on many platforms, but at a kilobuck a pop, it's probably not a personal purchase.

If your employer has "really deep" pockets, buy a Tektronix bitmap workstation. Smalltalk's included. And the Tektronix hardware and Unix port have been designed for Smalltalk, so it's better integrated.

If you're running on a PC clone or a Mac II, then Smalltalk-V is your best bet. It's inexpensive, well-supported, and the Mac II and 80286 versions are very fast. The environment, while differing considerably from Smalltalk-80, shares a large group of Smalltalk-80 classes. The version that runs on vanilla 8088 PCs is cheap, but lacks the speed and object space necessary for large applications.

Vendors

Digitalk, Inc., has sold more copies of its version of Smalltalk, called Smalltalk/V, than any other Smalltalk on the market. It runs on a number of platforms, and has been pared down especially for PCs. The version we bought came with 99 classes. It has very clearly written, nicely formatted documentation — some people would pay for the manual alone what we paid for the package, since you can't buy the manual without the software.

Smalltalk/V requires an IBM-PC, PC/XT, PC/AT, or compatible and PC-DOS or MS-DOS — costs $99.95. Smalltalk/V286 operates with the DOS operating system on AT class computers and requires a minimum of 1-1.5 MB of memory — costs $199.95. Digitalk's Macintosh Smalltalk/V runs on the Mac II, the Mac SE, and the Mac Plus; it will be available in the Fall — costs $199.95. There are applications kits available for color, communications, and other goodies — they cost $49.95 each. For more info —

Digitalk, Inc.
9841 Airport Boulevard
Los Angeles, CA 90045
1-800-922-8255
1-213-645-1082

ParcPlace Systems sells Smalltalk-80 for a number of platforms manufactured by Sun, Apollo, Hewlett-Packard and others, plus Apple Macintosh II, Plus, and SE workstations. See publications section for their address to write for specific information. Prices are over $1,000.

Tektronix was one of four companies that contributed to the development of Smalltalk-80 in the late 70s and early 80s. Apple, Digital Equipment Corporation, Hewlett-Packard and Tektronix were test sites for early versions of the system Xerox was developing — they each were expected to develop a full implementation of a written specification of the interpreter.

Tektronix has been developing its own implementation of the system since then, extending it from a version that Xerox PARC released in 1983. They have a very extensive class library, collection of tools, and reusable components. Tektronix Smalltalk is only available on Tektronix workstations.

A Little Smalltalk is a version of Smalltalk that runs under UNIX on line-oriented (conventional) terminals. The interpreter is written in C. It's available for distribution cost on 9-track tape in "tar" format. This Smalltalk doesn't support graphics (no windows, no mouse), but it's an inexpensive way to learn object-oriented programming. For more info —

Smalltalk Distribution
Department of Computer Science
Oregon State University
Corvallis, OR 97331

Periodicals

BYTE magazine, August 1981 issue. Entire issue devoted to Smalltalk-80. A collection of articles that provide an interesting and authoritative introduction to Smalltalk.

HOOPLA! (Hooray for Object-Oriented Programming Languages!) is published quarterly by Object-Oriented Programming for Smalltalk Application Developers Association (OOPSTAD). Annual individual membership in the U.S. is $25.

HOOPLA!
P.O. Box 1565
Everett, WA 98206-1565, USA
electronic bulletin board, 300/1200 baud: 1-206-252-9048

Journal of Object-Oriented Programming (JOOP) is published bimonthly. U.S. annual individual subscription is $49.

JOOP
P.O. Box 968
Fort Washington, PA 19034
1-800-345-8112

SCOOP is a newsletter published by Digitalk, Inc., for their Smalltalk/V customers who've sent in their signed warranty card. It contains new product and upgrade information, bug fixes, programming tips, and other news.

The *Smalltalk-80 Newsletter* is published by ParcPlace Systems (PPS), Xerox Corporation. It focuses on new PPS products and services and applications written in this vendor's Smalltalk-80.

ParcPlace Systems
2400 Geng Road
Palo Alto, CA 94303
1-415-859-1000 (in CA)
1-800-822-7880

theActiveView is a quarterly newsletter available free of charge to Tektronix Smalltalk customers. It focuses on features of Tektronix' implementation of Smalltalk-80, programming tips column applicable to Smalltalks from all vendors, updates on documentation and training classes offered by Tektronix. Send a postcard with your name, company name, address and phone number to:

Software Productivity Technologies
Tektronix, Inc.
P.O. Box 500, M.S. 50-470
Beaverton, OR 97077
ATTN: theActiveView -mc

Books

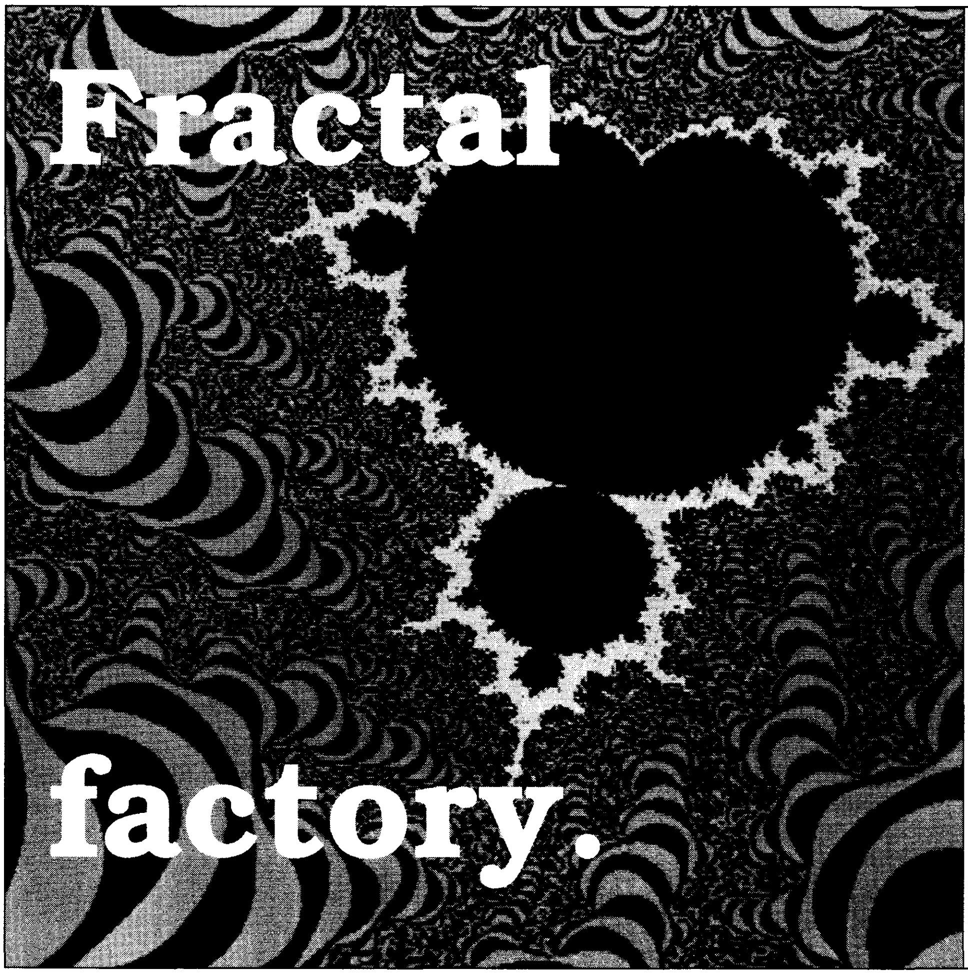
Budd, Timothy. *A Little Smalltalk.* Reading, MA: Addison-Wesley, 1987.

Goldberg, Adele and Robson, David. *Smalltalk-80: The Language and Its Implementation.* Reading, MA: Addison-Wesley, 1983.

Goldberg, Adele. *Smalltalk-80: The Interactive Programming Environment.* Reading, MA: Addison-Wesley, 1984.

Pinson, Lewis J. and Wiener, Richard S. *An Introduction to Object-Oriented Programming and Smalltalk.* Reading, MA: Addison-Wesley, 1988.

♦ ♦ ♦



A two–billion–power Mandelbrot/Julia microscope. Fast!
Aim-and-frame, quick draft, animate, recolor, retouch, multiple palettes, save and retrieve, cloud chamber, help and more.
Includes seven ready-made pictures for immediate gratification.

MANDELBROT EXPLORER 2.6

$30

Peter Garrison
1613 Altivo Way
Los Angeles, CA 90026
(213) 665 1397

16-color VGA/EGA to 800x600
Specify VGA or EGA, 1.2Mb
Overseas orders please add $4

Reader Service Number 112

Actor, An Object-Oriented Language

Taming Microsoft Windows

I'll try (real hard) to avoid the puns this time and simply set the stage. Actor is an object-oriented language written to run under windows.

When I first heard about Actor, I assumed it was a simple windows creator. I was wrong, very wrong. So I leave it to you to decide if Actor means curtains for the more classical performers.

Although object-oriented programming has existed for years on expensive dedicated computers, it's only recently reached PC users.

In this article I'll introduce you to object-oriented programming via a useful (real world) example — a critical path project manager. This example demonstrates several of the virtues of OOP (object-oriented programming), such as encapsulation and inheritance, and programming in the Microsoft Windows environment using my favorite language — Actor.

The application was easier to develop (and understand) using an object-oriented approach. I spent about two and a half man-weeks conceptualizing and programming, or about two-thirds of the time I would have spent using C, even though I have more experience with C.

In particular, I was able to reuse more code than I would have in C, and building the user interface was easier.

The project manager is similar to commercially available products such as SuperProject or MacProject, but on a smaller scale. It allows you to define a project as a group of related tasks and then compute the critical path of the project, its total cost, and so on.

The critical path is the sequence of activities that must be completed in a scheduled time to meet a project deadline. Some activities done in parallel may not be on the critical path, and thus have additional "slack time."

Editor's note: a standalone version of Zack's application that runs under Microsoft Windows, including complete source code, is available for downloading from the Micro Cornucopia RBBS, on the issue #44 disk, or directly from the author.

What Is OOP?

Unlike traditional languages where data is separate from the operations you perform on it, object-oriented languages connect data and functionality into modules known as objects.

Each object has a set of attributes (data) and operations. Objects can be "things" found in traditional languages such as arrays, strings, numbers, characters, files, etc. Actor includes a rich set of predefined objects such as stacks, queues, sets, dictionaries, windows, dialog boxes, etc.

Object-oriented Design

When designing an application in a procedural language such as C or Pascal, you'd probably begin by designing the data structures and then determining the functions you need. In object-oriented languages the data and functionality are combined in objects, so you don't consider programs in terms of routines that manipulate passive data. Instead, you think of a collection of active objects that operate on themselves.

So you need to determine which objects will make up the system, and the easiest way to do this is to develop a logical model of the system you're trying to create.

In cases where there is no clear logical model, you can determine the objects based on the user interface. For example, the objects you'd use for a spreadsheet would include the spreadsheet window and the cells.

The Project Manager

An approach you can take in the project manager is to set up a project object which is a network of nodes. Each node is an activity: either a task or a milestone. Tasks are activities which consume time and resources. Milestones are used to mark the completion of tasks.

For example, developing software is a task that will take time and money. Or you might have a milestone to indicate that the specs are written for a product and that it's time to begin programming and writing the user manual. The milestone itself doesn't take any time or have any cost; it just marks the completion of a phase in the project.

Since our application will run under Microsoft Windows, you also need to create a project window object that can draw a diagram of the network, handle menu commands, and so on.

If possible, you want to preserve the functional divisions in the logical model. For example, the project object shouldn't be concerned with details of how the total cost of a task is calculated; let the task worry about it! The only thing the project needs to know is how to update the total if the cost of a task changes.

Good object-oriented design encourages the creation of abstract data objects that clearly define public protocol and hide implementation details. Although this approach isn't required by Actor, I strongly encourage it since it can minimize maintenance headaches.

The object-oriented design approach requires more work up front, but generally pays off in the long run by providing better encapsulation than the traditional procedural approach.

Figure 1 lists some of the objects I'll use for the project manager.

Messages & Methods

An object is capable of responding to messages that are sent to it by other objects, by itself, or by the system. For example, you can print an object by sending it a print message. The code in Figure 2 illustrates some message sends.

By Zack Urlocker
The Whitewater Group
906 University Place
Evanston, Illinois 60201
312-491-2370

Figure 1 — Objects in the Project Manager Application.

```
    Network       a generic network of nodes with a start and end
    Node          a generic node capable of connecting itself

    Project       a network that knows the critical path method
    Activity      a node with an earlyStart and lateFinish
    Milestone     an activity that uses no time or resources
    Task          an activity that has time, resources and cost
    PERTTask      a Task where the time is estimated by PERT

    Resource      used by a task; has a name and cost

    ProjWindow    a window that can display a network diagram
    GanttWindow   a window that can display a Gantt chart

/***/
```

Figure 2 — Examples of Message Sends.

```
    print(A);                 /* all objects know how to print */
    draw(rect, hDC);          /* draw a rect in the display */
    cost := calcCost(P);      /* whatever P is, get its cost */
    reSize(aWindow, wp, lp);  /* the system tells us to resize */

/***/
```

Figure 3 — A Sample Method Definition.

```
/* This method defines the calcCost message for the Task
       class. Self refers to the task that receives the message.
       Time, resources and cost are instance variables defined
       for the Tasks.
       The cost is the fixedCost plus the variable cost of each
       resource used by the task. If the cost changes, send a
       message to the network to update its cost. */
    Def  calcCost(self | oldCost)
    {
      oldCost := cost;       /* store our old cost as a temp */
      cost := fixedCost;     /* starting value for the cost */

      do(resources,          /* loop through the resources */
        {using(res)          /* res is the loop variable */
         cost := cost + getFixedCost(res)
                 + getTime(self) * getVariableCost(res);
      });
      if cost <> oldCost then
        updateCost(network, cost - oldCost);
      endif;
      ^cost;                 /* return the new cost */
    }

/***/
```

Message sends are similar to function calls in procedural languages, with some important differences —

(1) The first parameter is the receiver of the message; the other parameters are arguments.

(2) The message makes no assumptions about the "type" of the receiver or the arguments; they're simply objects.

This provides a great deal of flexibility since you can have different objects respond to the same message in their own way. This is known as polymorphism (or literally, "many behaviors").

For example, you might have a calcCost message that can be sent to either a project or a task and, using calcCost's algorithm, will calculate the cost of the project or task.

The implementation of a message for a class of object is called a method, and it corresponds to a function definition in a procedural language. Figure 3 shows a sample method definition for calcCost for the Task class.

Actor source code looks a lot like Pascal or C. Most programmers find this makes it easy to learn. The method definition begins with the Def keyword followed by the name of the method, the receiver (always referred to as self), any arguments, and after a vertical bar, any local variables.

The do message is defined for all collection classes and lets us loop through the resources array referring to each element of the array in turn. If, later, you decide that the resources should be handled as a lookup table or a set, calcCost will still work correctly since all of these collections understand a do message.

Actor is an interactive environment. You write methods with an editor called a browser, compile them, and test them, as easy as one-two-three.

Classes Of Objects

Every object belongs to a class. A class is like a data type in procedural languages, but much more powerful. Classes define the data that make up objects and the messages that objects understand.

For example, a stack class (actually called OrderedCollection) includes instance variables firstElement and lastElement and responds to messages such as push and pop.

Instance variables are attributes of an object and are like fields in a record structure in C or Pascal. You can create new classes and define methods using the browser.

Rather than access the private instance variables of a class directly using the dot notation (e.g., stack.firstElement) you can encapsulate the data by using messages only. That way you reduce the dependencies on the implementation.

For example, if you need to get the time required for an activity x, you should send a getTime(x) message rather than refer directly to the instance variable x.time.

It doesn't matter if x is a Milestone, a Task, a PERTTask or even a Project, as long as it knows how to respond to a getTime message. This can make the critical path recalculation algorithm easier to write since you eliminate special cases for Milestones; you just define its getTime method to return zero. This results in significant savings in code.

Inheritance

What really makes classes powerful is the use of inheritance. You can create descendant classes that build on the functionality already found in the system.

For example, the OrderedCollection class mentioned earlier descends from the Array class. Similarly, you can create descendants of classes like Window and Dialog to build the user interface to this application without having to know the more than 400 Microsoft Windows function calls. Descendant classes automatically include much of the generic windowing behavior that users expect in a Windows application.

Inheritance lets us reuse code when something is "a little different" from an object that already exists. For example, I defined the classes Network and Node to provide generic network connection capabilities. You can create a new network, connect some nodes to it, disconnect nodes and so on.

This may not be very exciting, but it lets us factor out part of the application, test it, and then forget about it. Since there's nothing inherently network or node-like in the system already, these classes descend directly from class Object.

Figure 4 — The Connect Method.

```
Public protocol :

     connect(N1, N2);      /* e.g. N1 -> N2 */

  Private implementation:

     /* Connect self->aNode by updating self's outputs and
        aNode's inputs. Also, aNode should know it's network
        and the position should be set. */
     Def  connect(self, aNode)
     { addInputs(aNode, self);
       addOutputs(self, aNode);
       setNetwork(aNode, network);
       setPosn(aNode, self);
     }

     /* To connect node n1->n2, n1's outputs must contain n2. */
     Def  addOutputs(self, aNode)
     { add(outputs, aNode);
     }

     /* To connect n1->n2, n2's inputs must contain n1. */
     Def  addInputs(self, aNode)
     { add(inputs, aNode);
     }

/***/
```

Rather than keep track of the connections in the network, the nodes themselves maintain a set of input nodes and output nodes as instance variables inputs and outputs. The network, then, just maintains the start and end nodes. The network is actually implemented as a doubly-linked list, but this is "private" information and is encapsulated with connect and disconnect methods.

You can implement the connect method (see Figure 4) as addOutputs and addInputs messages since both the nodes must know about the connection that is made. These two steps could easily have been done in the connect method, but by having each node manage its own instance variables, you can deal with networks made up of other networks.

The setPosn method is another example of private protocol. I use this method to update the onscreen position of the node and I wrote the disconnect method in a similar fashion.

Interactive Tools

Once you've defined the network and node classes, you can create a network and examine it to make sure it's as you expect. Actor encourages an interactive programming style and has a rich set of tools to make it easy to test and debug code one piece at a time.

If any runtime errors show up, a source code debugger will pop up automatically. You can fix the code on the fly and resume execution.

Any time you want to examine an object, you can call up an inspector and see the values of the instance variables. You can also inspect the instance variables of an object from within another inspector and trace through an entire network easily.

Actor also has a code profiler that you can use to determine which methods will give you the most optimization. You can then selectively use early binding techniques to speed up critical code by eliminating the message send overhead. With the proper use of early binding in critical areas, you can improve performance by 25 to 30%.

The Project Manager Classes

Once you're sure the generic Network and Node classes are working properly, you can define the Project, Activity, and other classes listed in Figure 2.

The Project class will descend directly from the Network class and include additional functionality related to the application. For example, you need to be able to recalculate the critical path of the project.

The Activity class is a formal class that will group behavior common between its descendants Milestone and Task. You won't ever create Activities themselves (except for testing), instead they'll always be either a Milestone or a Task.

Alternatively, you could have had Task descend from Milestone, but you'd

DICTIONARY

A **Dictionary** is a collection of data pairs, much like an English language dictionary is a collection of term/definition pairs. The "term" entry is the key and the "definition" entry is the value of the pair, or association.

A Dictionary functions like a single-key database. This is how you create an object of class Dictionary with room for two key/value pairs:

```
   Worker1  :=  new(Dictionary,  2);
```

Next fill the dictionary with strings as keys like "Name" and "Age" and the corresponding values, the string "Sam Jones" and the integer 22. You are free to mix data types.

```
   Worker1["Name"]  :=  "Sam  Jones";
   Worker1["Age"]  :=  22;
```

Like a database, an Actor dictionary grows when you add to it more entries than you originally specified. Below, the dictionary grows from 2 to 4 as you add two more entries, or pairs:

```
   Worker1["Department"]  :=  #Engineering;
   Worker1["Reviews"]  := #(85  73  98  94  100);
```

Mixing data types allows great flexibility. You can even incorporate the above dictionary, Worker1, as a value in an entry of another dictionary, as in Employees below:

```
   Employees  :=  new(Dictionary,  10);
   Employees["Sam"]  :=  Worker1;
```

Access the data in a dictionary by specifying the appropriate keys. The following statement returns the array #(85 73 98 100):

```
   AWorkersReview  :=  Employees["Sam"]["Reviews"];
```

You can also retrieve data from dictionaries by enumerating over the dictionary and extracting particular entries. Here is how to create a separate "database" of employees working in the Engineering department:

```
   Engineers  :=  extract(Employees,  {using(employee)
      employee["Department"]  ==  #Engineering});
```

Another feature allows you to create a new database incorporating only a subset of the original's data. You do it by enumerating over a dictionary and collecting particular data about every entry:

```
   MailingList  :=  collect(Employees,  {using(employee)  employee["Address"]});
```

Dictionary objects are available in Actor as ready-made units of functionality. They can be modified or specialized for any application. However, as programmers, you are never involved in a dictionary's physical implementation or memory management.

Technical Specifications

- Runs with Microsoft Windows 1.x, 2.0 and 386.
- Pure, single-inheritance object-oriented language, incrementally compiled.
- Dynamic linking to C, Pascal, Assembler, or Fortran libraries. Pass data in C structures.
- Pascal and C-like syntax.
- Programming tools: Browser, Inspector, Debugger, File Editor.
- Full access to MS-Windows systems calls, multitasking, and DDE.
- Fast device-independent graphics: lines, shapes, icons, cursors, bitmaps, metafiles, Turtle graphics, sample control language using YACC.
- 150 classes, 1500 functions, fully extensible.
- Window styles: tiled, overlapping, popup, child, edit, dialogs. Controls: list boxes, scroll bars, buttons, check boxes.
- Data structures: stacks, arrays, queues, lists, dictionaries, sets, sorting, hashing, intervals.
- AI support: frames, symbols, dictionaries, lists, symbolic programming, functional arguments. Parsing and lexical analysis YACC compatible.
- String manipulation: substring, concat, append, insert, remove, search.
- 643-page manual includes tutorial and reference.
- No license fees. Generates stand-alone applications.
- Fastest interactive OOL available.

Prices

*Actor $495 • Academic $99 • Academic site license $99 • Manuals for site license $35 • **New!** Language Extension I $99 • Shipping $5 US, $25 Int'l*
***Tech Support**: Level 1 (20 phone calls) $100 • Level 2 (Developer's package) $250*
***Training Courses**: 201 Intro. to Actor and OOP (2 days) $395 • 202 Actor Developer's Workshop (3 days) $795 • 203 Both 201 and 202 (1 week) $995*

ACTOR®

Class of the Month
Vol. 1
No. 1

Actor: an interactive object-oriented programming environment running under Microsoft Windows.

Class: a template for data structures, or objects, with similar properties. These objects are tested, refined and reusable units of functionality you can snap into your Actor programs at any time.

The Whitewater Group®
Technology Innovation Center
906 University Place
Evanston, Illinois 60201

For more information call: (312) 491-2370

Reader Service Number 116

then inherit inappropriate behavior (certainly not good!).

Similarly, you could define a class called PERTTask that descends from Task that uses the Project Evaluation and Review Technique (PERT) for estimating time. The only methods you'd have to write would be those related to calculating time; everything else would be inherited.

By using inheritance you reduce the amount of code you need to write and the amount of testing. Once a method is written for the Activity class, you know it will work for any descendant classes. Figure 5 shows a class tree of the classes in the application. Note that all classes descend from the Object class.

The Recalc Algorithm

The user can set automatic recalculation to recalculate the critical path any time a change is made; or alternatively he or she can turn off automatic recalculation, make several changes, and then recalculate the entire network.

The critical path recalc algorithm must make two passes: a forward pass, recalc1, used to compute the earlyStart times of activities; a backward pass, recalc2, computes the lateFinish time and the slack. When an activity's time changes, it sends a recalc1 message forward to all of its outputs, who send it to their outputs, and so on.

Like today's popular spreadsheets, the project manager uses a minimal recalc algorithm so that only parts of the project that actually change are recalculated.

The recalc1 message is only passed on if the earlyStart or time changes, otherwise the outputs don't need to be recalculated.

Figure 6 shows the critical path recalculation algorithm.

Windows Objects

Of course, it'd be nice to have pulldown menus and graphic displays of the results. Since Actor is a Microsoft Windows development environment, these are easy.

Microsoft Windows is already somewhat object-oriented. When the user clicks on the mouse or presses a key, MS-Windows will send a message to the appropriate window. Because of the object-oriented approach, programming Microsoft Windows is much easier in Actor than in a procedural language like C. Actor has predefined classes for windows, dialog boxes, scroll bars, etc.

You'll define a new window class, called ProjWindow, which inherits the

Figure 5 — The Class Tree Diagram.

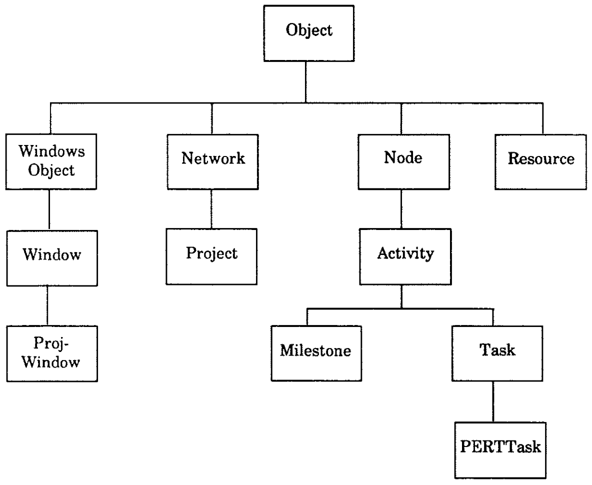


Figure 6 — The Critical Path Recalculation Algorithm.

```
Network class methods:

    /* Recalculate the entire network. Invalidate all nodes
       so that a full recalc will be forced. Then do both the
       forward and backwards recalc passes. The forward pass
       must be completed before doing the backwards pass. */
    Def  recalc(self)
    { invalidate(start, new(Set,10)); /* clear all nodes */
      cost := 0;                      /* recalc cost */
      recalc1(start, true, true);     /* force entire recalc */
      recalc2(self);                  /* backwards pass */
    }

    /* Do the backwards pass only starting from the end node.
       The end node is always critical. */
    Def  recalc2(self)
    { recalc2(end);
      end.critical := true;
    }

  Activity class methods:

    /* Recalculate the network from this node onwards.
       This requires forcing a forward recalc1 and a
       backwards recalc2 from the end of the net. */
    Def  recalc(self)
    { recalc1(self,true,nil);  /* force forward recalc1 */
      recalc2(network);        /* do backwards recalc2 */
    }

    /* Recalculate an activity's earlyStart. If the user has
       set an earlyStart, use it instead of recalculating.
       Send a message to the node's outputs to recalculate only
       if a forced recalc is required, or if earlyStart changes.

       formula:  ES = max(ES(i) + time(i)) for all inputs i

       arguments:
         timeChanged:  force a recalc1 if the time has changed
         costRequired: force a calcCost(self) if true
    */
    Def  recalc1(self, timeChanged, costRequired |oldEarlyStart)
    { if (costRequired)
        calcCost(self);
      endif;

      oldEarlyStart := earlyStart;         /* temporary */

      if (userEarlyStart)                  /* user over ride */
        earlyStart := userEarlyStart;
```

continued on page 24

It's Our Objective To Keep You Oriented

Get the latest insights and developments in Smalltalk, C++, Objective C, Eiffel, LISP, and Simula.

Innovative and applicable—tracks and evaluates the latest advances in this methodology.

Featuring:
Incisive peer-reviewed articles on:
- problem solving techniques
- reusable OOP components
- latest languages
- applications in artificial intelligence
- software maintenance
- program testing

Info-packed articles written by the best known researchers and premier software developers. Here's a sampling:

"Using Objects to Package User Interface Functionality"

"Designing Reusable Classes"

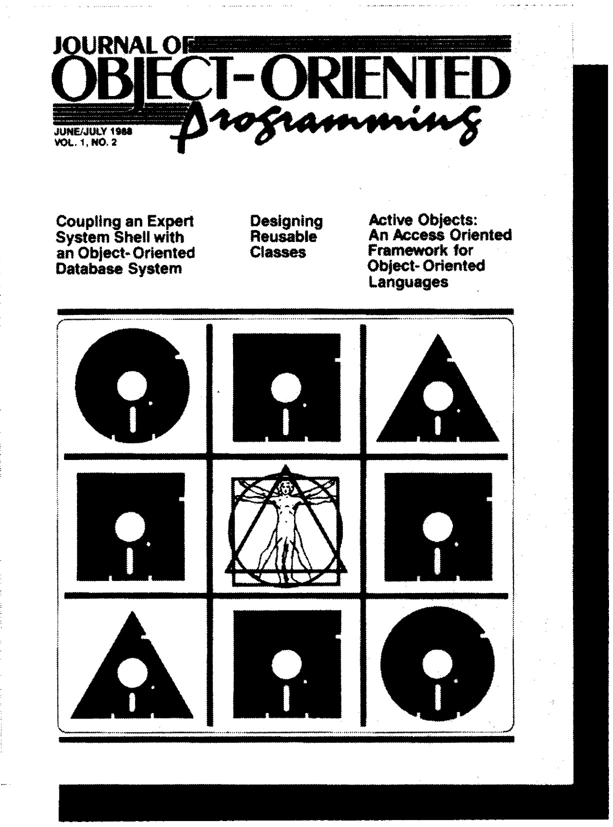
"Active Objects, An Access Oriented Framework for Object-Oriented Languages"

"Encapsulation, Reusability and Extensibility in Object-Oriented Programming Languages"

Get unbiased product reviews and up-to-the-minute product news. And personal interviews with the world's top programmers.

Written for software developers, programmers, engineers and computer scientists. Get original research that's definitive, authoritative and applicable. Get objective coverage that's enlightening, reliable and functional. Subscribe today!

THE JOURNAL OF OBJECT-ORIENTED PROGRAMMING



JOURNAL OF OBJECT-ORIENTED Programming
JUNE/JULY 1988
VOL. 1, NO. 2

Coupling an Expert System Shell with an Object-Oriented Database System

Designing Reusable Classes

Active Objects: An Access Oriented Framework for Object-Oriented Languages

Order toll free (7 days a week) **1-800-345-8112**

JOOP

☐ 1 year (6 issues) $49
☐ 2 years (12 issues) $89
Outside the U.S. add $24/yr air service.
Check enclosed ☐ Bill Me ☐
Charge my: VISA ☐ Mastercard ☐

Card # ______________________ Exp. Date ____________

Signature ______________________

Your Name ______________________

Company ______________________

Address ______________________

City ______________________

State ______________________ Zip ____________

Mail to: **JOOP,** 310 Madison Avenue, Suite 503, New York, NY 10017, or send for a sample copy. 1 88

TINY EINSTEIN

The Expert System Shell

- Create your own expert systems in minutes.
- With pulldown menus and windows
- Context-sensitive online help
- Free example expert systems
- Tutorial
- Interactive full-screen text editor
- DOS access from shell
- Turbo Fast execution
- Cluster, Trace, Explain
- For Diagnosing . . .
  Simulating . . .
  Predicting . . .
  Planning . . .
  Classifying . . .
  Training . . .
  and Monitoring systems.

Only $49.95! (Plus $5 S/H)

Reader Service Number 72

Designed & implemented by
Gary Entsminger & Larry Fogg

ACQUIRED INTELLIGENCE
P.O. BOX 2091 • DAVIS, CA 95617 • (916) 753-4704

"default" behavior needed in Windows applications. Note that ProjWindow and Project are unrelated in the class tree. ProjWindow does not descend from Project since it's not project-like. Rather, a ProjWindow simply contains a project object as an instance variable.

By having two distinct classes, you clearly separate the user interface issues from the recalculating of the critical path handled by the project and its activities. This makes it easy to have different types of windows display projects in different ways.

ProjWindow can display a network diagram of the activities while a GanttWindow will display a time line graph known as a Gantt chart. In both cases, the windows will send messages to the project to select particular activities, edit them, recalculate the critical path and so on.

Conclusion

Object-oriented programming encourages the creation of abstract data objects with a well-defined public protocol and a private implementation. The result is a program that's easier to develop and maintain.

The project manager is a good demonstration of object-oriented principles put into action. And with its graphics, fast recalculation, file load and save capabilities, it's downright useful.

♦ ♦ ♦

Continued from page 22

```
else
     earlyStart := 0;                  /* find largest ES */
     do(inputs,
       {using(input)
        earlyStart := max(earlyStart,
                    getEarlyStart(input) + getTime(input));
      });
    endif;

    /* Recalculate outputs if earlyStart changed OR if time
       changed. Don't force it to beyond the next level. */

    if timeChanged or (earlyStart <> oldEarlyStart)
      do(outputs,
        {using(output) recalc1(output, nil, costRequired);
      });
    endif;
  }

  /* Recalculate an activity's lateFinish then its slack and
     determine if its critical.
     If the user has set a lateFinish, use it instead of
     recalculating. LateFinish recalc goes backwards from a
     node to its inputs. Always propogate recalc2 since a
     change to the time of a node will not change lateFinish,
     but it can change slack and critical, which are only
     known on the backwards pass.

     formula:  LF = min(LF(i) - time(i)) for all outputs i
  */
  Def  recalc2(self)
  {
    if userLateFinish
      lateFinish := userLateFinish;  /* user override */
    else
      lateFinish := MAXINT;          /* find smallest LF */
      do(outputs,
        {using(output)
        lateFinish := min(lateFinish,
                    getLateFinish(output) - getTime(output));
      });
    endif;

    calcSlack(self);
    calcCritical(self);

    /* Continue sending the recalc2 message. */

    do(inputs,
      {using(input)
       recalc2(input);
     });
  }

/***/
```

Koala

KOALA COMPUTER, INC.

213-316-5866
9 to 6 PST M - Sat

30-Day Money Back Guarantee
Software sales are final except replacement of defective media. Merchandise must be undamaged for full refund.

!! VGA !! ATI VIP card and VGA Professional Color Monitor — $648

Analog 13" anti-reflective, non-interlaced, flicker free, .31 dot pitch monitor with an ATI improved VIP video card. 800x560 drivers for ACAD, Windows, GEM, and Ventura Publisher included. VGA, EGA, CGA, MDA, HGA selectable. Up to 256 colors on screen at one time with VIP card. 37 lb.

11½"x 14⅞" Greenbar Paper with Carbon Paper - 45 lb. $15

2 part 1500 sets, 4 part 700 sets. Two holes punched for binders. Case approximately 45 pounds. Send self-addressed stamped envelope for free sample.

SUPERIOR XT* STYLE KEYBOARD for the TOUCH TYPIST - $25

Replacement XT* style keyboard with LED in Num Lock and Cap Lock keys. Manufactured by **CHERRY**. Complete with coil cord and plug ready to slip into your case. Our service department uses these when possible instead of repairing the old one as most people appreciate the nice feel of these **made in USA** CHERRY keyboards. 2 lb.

MONOCHROME DISPLAY ADAPTER with PARALLEL PORT - $35

Japanese mfg. board. This is the MDA that can be plugged in with a CGA (Color Graphics Adapter) at the same time. 2 lb.

384K EXPANSION WITH IO and RAM - $139

Parallel, serial, game, clock calendar, and 384K of memory installed and tested. **$139 3 lb.**

NO SCREWDRIVER REQUIRED PARALLEL PRINTER CABLE - $10 - $16 - $20

Not the cheap ones but the DB25 with the knurled fastener. **6 foot** $10 one lb., **10 foot** $16 two lb., **15 foot** $20 3 lb.

The COMPLETE DBase* SYSTEM for the SMALL BUSINESS - $4

DBase* command code and operational instructions are contained in this excellent 335 page 8½"x 11" book DBase APPLICATIONS in BUSINESS 2 lb.

SURGE SUPPRESSOR 6 OUTLET POWER STRIP with EMI/RFI FILTER - $15

Full 3 line protection in normal and common modes. Noise filter helps protect against noise interference. "Push to reset" circuit breaker protects against overloads. 6 foot 14/3 SJT power cord. On/Off switch. Continuous surge protection for your valuable electronic equipment. UL listed. 3 lb.

***TERMS** - Cashiers check, immediate shipment - All others must clear. No COD. Prices FOB Torrance, CA. Call us and give your ZIP code and we will quote freight charges. California residents add 6½% sales tax. Add $2 handling to orders under $25.*

KOALA COMPUTERS INC.

4306 Torrance Boulevard, Torrance, California 90503 (Offices only)
CALL 213-316-5866 9 to 6 PST M-SAT

*XT trademark IBM, DOS - Microsoft, DBase - Ashton Tate

Reader Service Number 88

Thinking Objectively

A Biological Approach To Designing Neural Networks

Computers are dumb. Right? They only do what you tell them. Right? Even expert systems boil down to a list of rules, and that's about as fancy as computers can get. Right?

Wrong. (You knew I'd say that.)

If Doug and Arthur are correct, we'll just teach computers how to learn and then turn them loose. In no time at all, they'll be the experts. All by themselves. Mankind can go back to manual labor.

Our minds have the (as yet) machine-unsimulatable ability to make sense out of noise. We can distinguish individual (and often subtle) meaning out of a cacophony of sounds.

Imagine a symphony. You not only hear the orchestra (as a whole), but also distinguish many of the individual instruments. You can even hear people whispering in the audience or the siren of an emergency vehicle outside.

You differentiate among sounds although they occur simultaneously, have overlapping frequencies, and are input through only two analog channels — your ears. In your mind, different sounds represent different "objects."

The ability to recognize objects is essential for high-level intelligence and must precede, to some degree, any high-level association of meaning.

If a squirrel, for example, repeatedly sees a cat and learns that the cat is dangerous, it *first* recognizes the cat as a separate object in the real world. Then it associates the object, cat, with danger.

The recognition step is essential. Without it the squirrel would have no quick way of associating danger — danger would be entwined with an infinite number of real world stimuli. The learning of objects is fundamental to intelligent organisms. And it's of equal importance in the design of intelligent machines. Despite this, we don't have a good understanding of the idea of object.

For the past three years, we've been building systems that use conventional and electronic technology to explore (in depth) the idea of the object. The results are encouraging.

We believe that a better understanding of what the object is will remove a longstanding roadblock in the transition from fixed design methods (e.g., expert systems) to the development of truly automatic learning technologies.

In this article we'll present some of the fundamental concepts involved with object recognition and some general approaches to implementation.

Who Teaches?

Many traditional neural networks can learn to recognize stimuli — but these networks must be told what they're learning. We do this by telling the network to learn when the desired stimuli are present. In other words, we teach the network.

We decide what the network will learn by placing chosen stimuli (objects) in nonconflicting scenes, and directing the network to respond when it recognizes a particular set of stimuli. Is this how animals learn?

We don't think so. Although you might argue that animals learn about danger from their parents or from their genes, it seems more likely that the world, itself, contains the clues needed by animals to group information. So far, neural nets have not tried to automatically extract this information.

Seen It Before; See It Again

Most of us make the erroneous assumption that neural tissue "takes in" data, puts it somewhere, remembers where to retrieve it, and reproduces the data intact. Conventional computers operate this way, but neural tissue doesn't. It neither stores input data nor needs to!

How can tissue remember (recognize) objects if it hasn't stored data?

Good question. When remembered stimuli appear at the input of neural tissue, the tissue responds, that is, it determines whether it knows (has experienced) the stimulus.

In other words, if the tissue has experienced a stimulus or group of stimuli before, it can experience it again. See Figure 1.

Animal Learning

There are many animals, some very complex, that appear to learn little (or nothing) in their short lifetimes. The highly stereotyped responses of many insects fall into this category. How do these animals acquire their useful behaviors?

When we realize that life, in general, is a learning system, we realize that the design of these insects (their genetic makeup) was learned through repeated testing and gradual improvement. The result is similar to a well-tested expert system that's been given the necessary complexity to be successful.

But many animals (like people) seem to learn throughout their lifetimes. They absorb knowledge of an extremely complex real world and automatically adjust to deal with it. It's known that we have over 10,000,000,000,000 neural synapses and only about 100,000 genes, or developmental instructions, to encode these connections.

It's obvious that these are far too few genes to supply enough information for the design of such high-level order. The genes, nevertheless, provide the necessary framework to get learning under way, although most of the complexity of life is acquired (learned) thereafter through experience.

As the designers of artificial intelligence systems, we might consider a similar approach. A fixed design would work in many simpler cases, but we'd have trouble deliberately designing-in the

By Doug Gaffin & Arthur Gaffin
Dept. of Zoology
Oregon State U.
Corvallis, OR 97331

Neuro Dynamics
1514 Canna Court
Mt. View, CA 94043

complexity of a high-level intelligent system. A modified form of mammalian intelligence may actually be the best approach. If we design and build the framework, the rest of the system can be self-learned.

The Forces Of Weak Association

Within real world experiences, there are relationships between events that provide hooks for learning. We call these relationships "the forces of weak association."

The term weak is used because these relationships are useful only in a statistical sense — they don't provide immediate absolute associations. These observations are simple and very important. They give intelligent animals the ability to gradually develop a correct and complex model (not a copy) of nature.

The forces of weak association are —

(1) If a stimulus is similar to another stimulus, not necessarily occurring at the same time, then the associated events in nature are likely to be related.

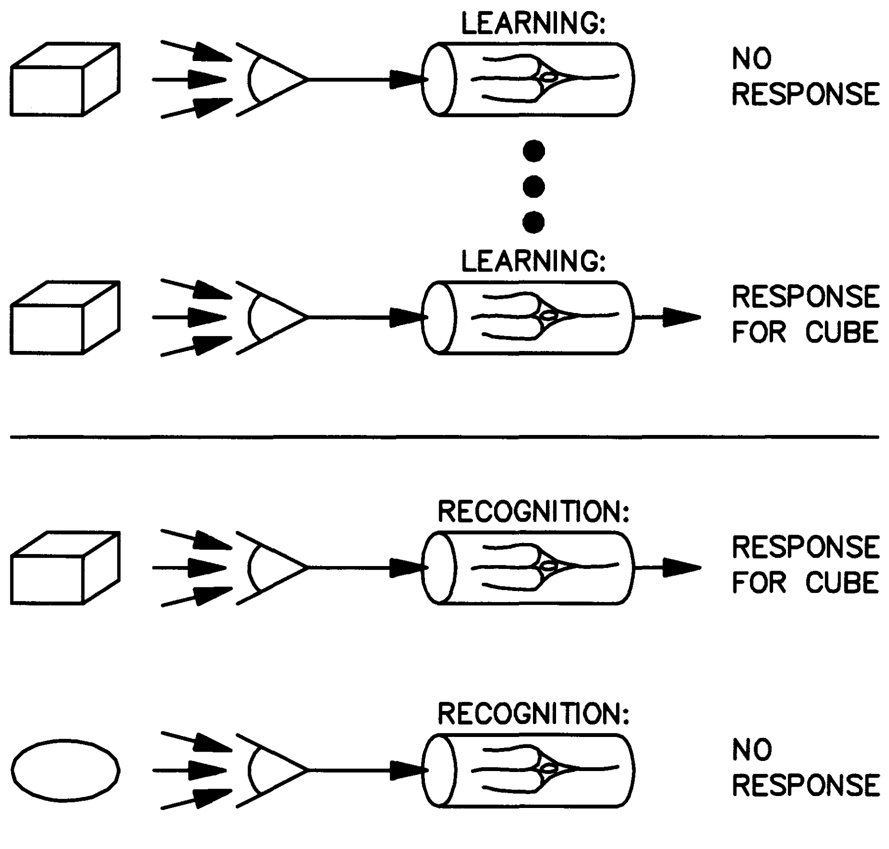
(2) If two or more stimuli are repeatedly encountered proximate with each other, in terms of time and/or space, the events are likely to be related.

(3) The more proximate the observation, the more important the relationship.

(4) The more frequent the observation, the more important the relationship.

(5) Most events in nature are continuous. For example: If an object is detected at one moment, it is likely to remain present for an extended period.

Figure 1 — Simple Learning



The All-Important Object

Let's look at the concept of an object in more detail. Suppose we show Rob, our robot, a coffee cup that's resting on a table. We want Rob to learn to recognize coffee cups. To make this task more interesting, assume that the cup casts a shadow.

If Rob is to learn what a coffee cup looks like, he must learn where the cup ends and the table begins. And he must recognize that the shadow isn't part of the table.

How do we recognize shadows, cups, and tables? By learning their intrinsic qualities during many experiences of cup, table, shadow.

We see the cup in different locations and circumstances on the table. We observe the stimuli that relate to the cup, say the handle, to be more frequently with the rest of the cup. And we observe stimuli that more frequently relate to the table, cup or no cup. The shadow is really no different; we observe it most frequently with the cup. Therefore, the shadow eventually becomes part of the cup object.

With this in mind, let's define an object —

An object occurs when certain patterns of stimuli repeat themselves, in a manner that's similar, in terms of time and/or space, with a frequency of repetition that is, on the average, relatively high.

Or, more simply —

An object occurs when two or more stimuli are observed repeatedly together. The real world is ordered (we order it!)

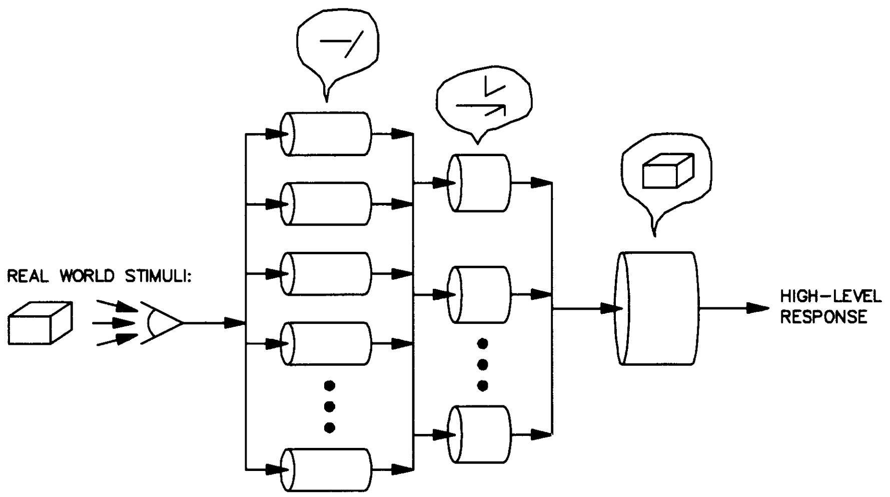
through a multitude of associations. Objects represent the embodiment of this order — forming the basis of fundamental associations.

Objects Yield Hierarchies of Knowledge

Suppose we have a piece of neural tissue and we program it (give it the ability) to learn and recognize frequent combinations of stimuli. We also program it to respond uniquely to each of these learned patterns of stimuli.

This is a high frequency filter — emerging responses correspond only with frequently occurring events at the input. Infrequent events aren't remembered and, therefore, aren't passed to the output as responses.

Figure 2 — Hierarchies



The events that are learned by this piece of tissue are objects. The organism takes the world for what it is and extracts information from it to build its model.

An infant in its first days of visual experiences can't physically alter its visible world, except for the direction and quality of view. The initial flow of data is largely one-way, from the real world to the infant.

The most frequent of these patterns of stimuli will be learned first and these will consist of small relationships that repeat themselves frequently — long edges, corners, basic shading patterns, types of blobs, etc.

The next level of learning will be in terms of these low-level features. Since the input stimuli of the next level are the responses from the first level, the next level sees its real world through the filters of the first level.

A third level may be added in the same manner, and so on. With many levels, very general and complex objects can be learned and recognized. See Figure 2.

Tricks Of Tissue

How can we design a simple, fast, and self-adjusting mechanism that automatically learns to sort out the most frequent stimuli? Let's look at a piece of tissue at one level in the hierarchy. How can it remember to recognize only stimuli that are frequently observed and therefore important?

Studies suggest that neural tissue employs temporary memory in the synapses (points of connection between neurons). These synapses are eventually converted to permanent memory if repeatedly reinforced. Some studies also suggest that neural growth may be affected by the level of activity the neuron.

Also, the growth of some neurons seems to be affected by the activity of adjacent neural processes. It's therefore possible that neurons may be selectively seeking other neurons with high levels of firing activity with which to form connections (synapses).

Uninvolved neural processes can be allowed to forget by remaining unconnected, being routed around (overwritten) and, in certain cases, atrophying. Biological systems have developed other tricks as well to selectively filter only the most frequent of stimuli.

Trick 1: Crowding

By simply "crowding" the tissue, we can create a competition for synapses. Less frequent stimuli will be reinforced

DBASE™ for Norton Guides™
INTRO PRICE $69.00

DBASE *ON LINE* is a "pop-up" DBASE language reference system which includes over 2 million bytes of complete reference with clear concise descriptions, and detailed examples to every command function and feature for:

CLIPPER™(Summer '87)	**QUICKSILVER**™(Diamond Release)
dBASE III Plus™	**dBXL**™(Diamond Release)
dBASE IV™	**FoxBASE+**™

DBASE *ON LINE* is powered by the Norton Guides "reference engine":
- Memory requirements: maximum 65k.
- Can run in resident or pass-through memory mode.
- Can be "popped-up" any time, inside any program.
- Automatically looks up a keyword read from the screen.
- Full or half-screen display.
- All available Norton language guides will run under the Norton "reference engine".
- Also included is a compiler and linker, allowing the creation of your own reference guides.

Additional Features:
- Tables - keyboard return codes, line drawing characters, color codes, error codes, ASCII chart and much more.
- The Clipper and Quicksilver guides also include commands and switches for the compile/link cycle, a table of reserved words and complete reference to the extend system.

ORDER: DBASE *ON LINE* and get the above language guides, a reference engine, a reference guide compiler/linker and manual.

30 Day Money Back Guarantee • Satisfaction Guaranteed
To order, call or write:
SofSolutions
440 Quentin Dr. • San Antonio, TX 78201 • (512) 735-0746

Trademarks: Norton Guides/Peter Norton Computing, dBASE/Ashton Tate, Clipper/Nantucket, dBXL Quicksilver/Wordtech Systems, FoxBASE+/Fox Software

Reader Service Number 108

less often and tend to become overpowered. See Figure 3.

Crowding gives neural tissue a simple, automatic mechanism to reduce the complexity of real world data — it reduces the data to the most important (frequent) events and responds only to these. The next level can repeat the process using the output of the previous level, thereby automatically creating orderly hierarchies.

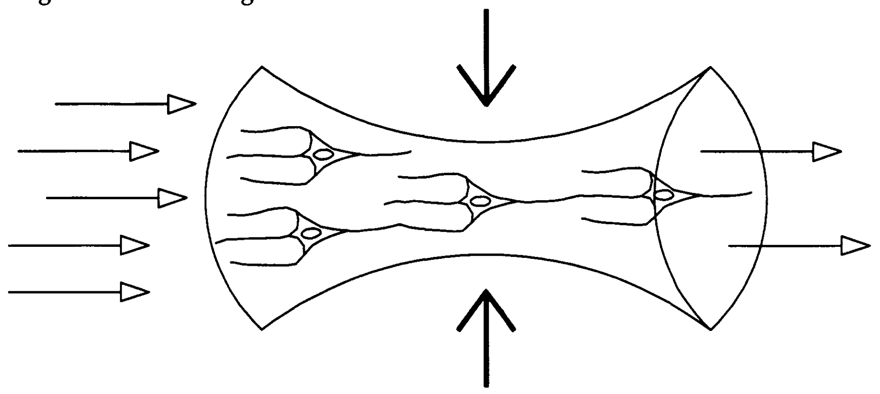
Trick 2: Inhibitory Synapses

Studies have shown that the connections (synapses) between neurons are inhibitory as well as stimulating by nature. When neural activity of inhibitory neurons reaches one of the places where two neurons touch, the pre-synaptic neuron inhibits activity in the post-synaptic neuron.

Since one neuron may contact as many as 10,000 other neurons, we can see that many neurons may be silenced (or at least partly silenced). This phenomenon can implement crowding even when there are more than enough synapses. It also causes active ideas in that tissue to choose addresses (synapses) that are as unique as possible for use as memory. Animal tissue, therefore, has implemented a clever, fault-tolerant addressing scheme that works even if some of the tissue is damaged.

Now that we understand why we need these inhibitory synapses, we can select whichever techniques work. It's important to note that we *don't need* to use the massively parallel approach once we understand what problem these tissue features are solving.

Figure 3 — Crowding



CROWDING is useful for learning and recognizing only the more important patterns of stimuli. Note that the pathwidth set of possible values of the input stimulus is greater than that of the response, thereby forcing a net reduction of the incoming data.

TRILOGY

A Powerful Procedural, Database, and Declarative Language.

SPEED — Where Prolog must backtrack, Trilogy can often solve the problem logically. Trilogy takes advantage of logic constraints (they constrain the search to possible solutions) which either eliminate backtracking or reduce millions of backtracks to a very few.

SYNTAX — Trilogy uses an intuitive, Pascal-like, program structure.

INTEGRATION — Trilogy is complete. It's the only language you need for writing Pascal-style routines, database handlers, and Prolog-style programs.

MODULARITY — Trilogy is modular language, very similar to Modula-2.

ENVIRONMENT — A complete programming environment, you get editor, library, linker, loader, error handling, automatic make, and contextual help. Plus, you get modules for: math, string handling, file manipulation, windows . . .

A TRUE COMPILER — Trilogy is an interactive compiler which produces native code for the 8086/8087.

LOGICAL PURITY — Trilogy was designed from scratch as a simple language with a completely logical foundation. Trilogy's speed results from its design, not from added commands. (Prolog's assert, cut, var, and retract, are not logical parts of that language. They were added to improve performance.)

DATABASE SUPPORT — Trilogy supports: variable size records, records with arbitrary values (lists, recursive trees); plus record insertion, deletion, and modification. (Anywhere in the file.) Files are relations and can be queried from within the language.

PRICE — Only $99.95 postpaid, U.S. funds. Plus $5.00 shipping & handling. Or $12.00 shipping & handling outside North America. Check, money order or VISA accepted.

Order From: **COMPLETE LOGIC SYSTEMS**
741 Blueridge Ave.
North Vancouver BC Canada V7R 2J5
(604) 986-3234

ONLY $99⁹⁵

Reader Service Number 71

Trick 3: Winners Tire

If you look at Necker's cube for 30 seconds (see Figure 4), it appears to flip back-and-forth between two opposing interpretations (ideas). We can speculate that the winning idea eventually fatigues and, therefore, allows the second one to win.

Eventually that one fatigues, and the first idea wins again, and so on. Since the first idea is still partially fatigued from the first time, it will fatigue more quickly this time. The flip rate eventually settles at about two times per second which is directly related to our neural fatigue rates.

Fatigue is neural tissue's form of a do loop — it provides a foolproof way of trying many alternative ideas before repeating the original one. It implements a tissue-level checkoff list that automatically moves from one idea to another.

Given a multi-level hierarchy with feedback at different levels, we can see that a search through combinations of ideas can be very complex and would never exactly repeat itself. It's likely that the subconscious mind uses some of these mechanisms — it would help explain why it takes a long time for some answers to pop into our mind.

Fatigue adds a self-adjusting quality to the system and causes it to perform better in a complex world. It also keeps the system from getting stuck in infinite feedback loops.

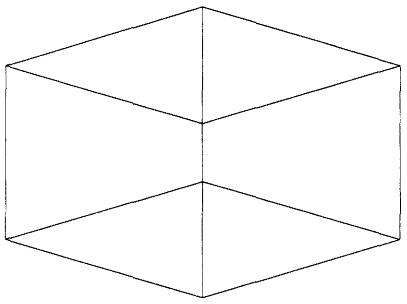
Since slowly changing stimuli tend to fatigue the associated paths, rapidly changing stimuli will have a competitive advantage — therefore fatigue created a bias towards changing stimuli. We can call this "tissue-level curiosity," the tissue actually gets bored with old information and responds better to dynamic situations.

Trick 4: The Axon

The real world provides an abundance of data, in fact so much data that the same exact set of stimuli or scene is never observed (experienced) more than once. We know that similar scenes are observed repeatedly, but the data in these similar scenes usually does *not have* many pixels of information that are the same. How, then, are these scenes similar?

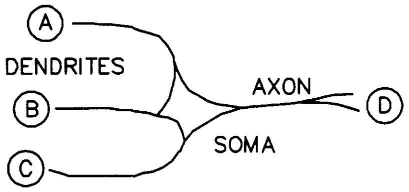
Let's consider a stimulus as a number which represents a point in a special highly multi-dimensioned space called, for lack of a better term, white space. This white space needs to be vast enough to represent a rich repertoire of stimuli, but not so vast that incoming stimuli cannot readdress previously learned points.

Figure 4 — Necker's Cube



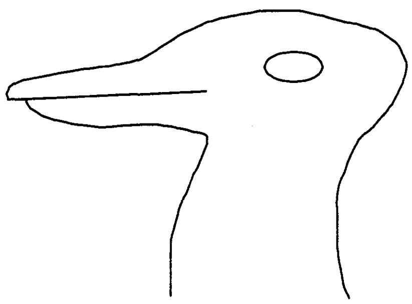
NECKER'S CUBE is an optical illusion that illustrates your neural tissue in action. When looked at for 30 seconds, the image will appear to flip back and forth between two different recognitions.

Figure 5 — Two out of Three Rule



The neuron acts as a digital reduction device. A simplified model could be a device that fires an action potential through the axon whenever at least 2 out of the 3 possible dendrites are active.

Figure 6 — Conflicting Images.



The drawing above should at first look like a DUCK. The drawing can also look like a RABBIT. Note that after the high-level idea of rabbit is thought of, all of the low-level features are immediately converted to those of a rabbit.

The neuron is ingeniously designed to provide an operational solution to this problem. A neuron consists of the following computational components:

- input processes - dendrite(s)
- decision mechanism - soma
- output processes - axon(s)
- inter-neuron connections - synapses

The neuron provides a very rich set of computational ingredients:

- COMPLEX INPUT system — inputs through the dendrites which provide a wide pathwidth of data.
- SUMMATION of the inputs as received by the central neuron body, the soma.
- DIGITAL DECISION system, in the soma, which acts somewhat like an analog to digital converter — it acts as a democratic element which fires only if enough inputs (votes) are present.
- DIGITAL OUTPUT system which fires and sends out an action potential through the axon.
- COMMUNICATION system in which the axon contacts many other dendrite processes of other neurons.
- MEMORY that is contained in the connection strengths of the synapses (the inter-neuron contacts).
- LEARNING which is the process by which the connection strengths and the locations of the synapses are formed.
- RECOGNITION when the system is excited by inputs and fires a digital output.

The neuron can receive many combinations of possible stimulus inputs which represent a fairly large white space. The output (axon), however, is binary (digital) — fire or no-fire.

Note that the neuron is functioning as a data REDUCTION device. By making decisions, the data is simplified and reduced by the action of the neuron. This process keeps the white-space of the data at any point in the neural tissue within operation bounds. See Figure 5.

Trick 5: Feedback

Some optical illusions show that the same picture (stimuli) can provide alternate interpretations. See Figure 6.

Some important points are listed below:

- Some illusions display alternate interpretations where only one can be visualized at a time — in other words, they compete.

- Many of the supporting low-level features are interpreted in concert with the overall interpretation.
- When the overall interpretation changes, these low-level features change also.

What is the usefulness of this mechanism?

- By the context of the overall interpretation, lower-level features can be more easily recognized, especially if the stimuli is difficult to interpret.
- If lower-level features were allowed to resolve totally on their own, they would seldom conclude a useful high-level recognition.
- Fatigue, ordinarily a very useful mechanism, would make matters difficult — the low-level recognitions would flip back-and-forth with little contribution to determining a valid high-level recognition. See Figure 7.

The subject of feedback extends far beyond the purposes discussed here — it's likely to be involved in nearly all neural processes in one form or another. The mechanism discussed here is fairly simple and easy to implement. In fact, we have implemented it in Turbo Pascal. The complete source code (and user-friendly .EXE) is available on the Micro C RBBS and from the authors.

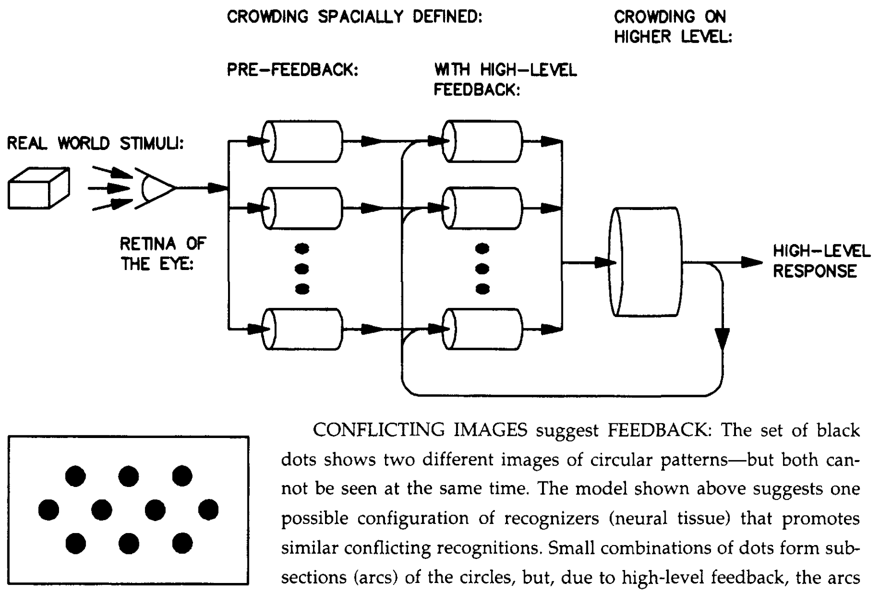
Summary

Electronic technology is moving forward at an extremely rapid rate, providing unprecedented computational power for the money. We now have more tools than ever for building neural network based systems. As expected, the level of public and private interest in neural intelligence is increasing dramatically.

With our new understanding of automatic learning, we are now ready to remove major roadblocks to our progress. By designing-in the ability to learn complex objects within complex hierarchies, we delegate most of the learning to the system — thereby making our job much easier.

♦ ♦ ♦

Figure 7 — Conflicting Images suggest Feedback


CROWDING SPACIALLY DEFINED:
CROWDING ON HIGHER LEVEL:
PRE-FEEDBACK:
WITH HIGH-LEVEL FEEDBACK:
REAL WORLD STIMULI:
RETINA OF THE EYE:
HIGH-LEVEL RESPONSE

CONFLICTING IMAGES suggest FEEDBACK: The set of black dots shows two different images of circular patterns—but both cannot be seen at the same time. The model shown above suggests one possible configuration of recognizers (neural tissue) that promotes similar conflicting recognitions. Small combinations of dots form subsections (arcs) of the circles, but, due to high-level feedback, the arcs are interpreted according to the context of the high-level idea (one circle or the other.

dBASE III+ 20 TIMES FASTER!!

DBDO

dBASE III+ Enhancment utility

"What a difference! No more watiting for output while I could have been processing other data. It's great!"
V. Kovacs
Penn Services

"I'm using it in every new system I write. Super for creating test data from large files. It's fast, easy to use, and follows dBASE syntax."
W. H. Whitney
McGraw Hill, Inc.

FAST:
Up to 20 times faster than dBASE.
In one case, report generation on a 60,000 record file was reduced from 18 hours to 2 hours!

FLEXIBLE:
Call from a program file or DOS prompt.
Run on a stand alone PC or a network.

Commands: COPY DELETE APPEND RECALL REPLACE COUNT MANY MORE

EASY:
dBASE-like syntax - No need to learn another language.

Functions: LTRIM[] UPPER[] TRIM[] SUBSTR[]

COMPATIBLE:
Recognizes and creates dBASE III+ files.
Transfer DBF data to DAT files for use with other languages (Basic, Pascal, etc.)

Introductory Offer
$99.00
Offer ends 12/31/88

FOR ADDITIONAL INFORMATION CALL: (215) 536-5858
Computerized Processing Unlimited
Country Square Shopping Center
Quakertown, Pa. 18951

Reader Service Number 105

Building MicroCad:

Design With C++

Okay, objectively, C++ might be interesting, but what can you do with it that you can't do just as easily with C? Bruce's MicroCad is a good example. It lets you create and manipulate graphics (objects). (This is the fourth in this issues trilogy of object-oriented articles.)

Before I introduce my tiny graphics program, MicroCad, I'd like to describe object-oriented programming. Three key features make a language object-oriented —

(1) abstract data typing, data encapsulation, or data hiding.

(2) type derivation or inheritance.

(3) commonality, polymorphism or, (in C++) virtual functions.

The first feature says you can create your own data types, with internal data and external operations. For example, you can create a complex number (a type) which contains a real and an imaginary part.

The C++ compiler will know how to deal with this new type's real and imaginary parts (which are effectively hidden), just as it knows how to handle the hidden exponent, mantissa, and sign parts of a floating-point number.

Your new complex type looks virtually no different to the compiler than a built-in type like double.

The second feature, inheritance, says you can derive a new data type from an old one (called "base"), instead of building new types from scratch every time.

The new data type is an old one with some additions and/or modifications. The parts of the old data type that work, you leave alone. Just change what you need to.

Feature three says you can control a derived data type through its base. This is very powerful, but takes some getting used to, since there's no equivalent in conventional programming languages. I'll go into this more later.

Each of these user-defined (or derived) data types is a class, the key feature which makes C++ more than just "a better C."

I'd like to illustrate these features by building a miniature CAD program (MicroCad) which lets you draw, manipulate, and delete shapes.

We'll create a generic base type called CADSHAPE, and from it derive CIRCLE, SQUARE and LINE.

Each derived type will know how to do some things, such as draw and delete itself. So you can create a list of CADSHAPEs and then step through the list telling each type (or shape) to draw itself.

Zortech C++

Before I go into the details of MicroCad, I want to briefly mention the C++ system I'm using for this example.

Zortech currently markets my C++ of choice. It's a $99 native-code compiler written by Walter Bright (who created Datalight C).

I became involved with this project back in the beta-test stage, wrote the introductory C++ chapters in the manual, and now am a technical support consultant. (In other words, I'm taking money from them so my opinions are tainted.)

The Zortech C++ package includes a C compiler, an integrated editing environment, a librarian, a nice make facility, along with mouse, graphics and standard C libraries.

You invoke the compiler with "ZTC filename," and it figures out from the file extension whether to use C++, C, or the assembler.

MicroCad requires Zortech C++ because I'm using their mouse and graphics libraries. Those of you who are graphics and mouse pros, bear with me — I've never messed with these things before, so I may do things which seem appallingly stupid.

But C++ actually saves me here, since all the low-level stuff is encapsulated — you can fix my flubs without changing the structure of the program.

OO & C++

Before getting into the core of the code, I want to say a few things about object-oriented programming in general, and C++ in particular.

A pure way to think about object-oriented programming (or OO) is that it consists of simply "sending messages to objects." This is the Smalltalk view: you tell an object *what* to do, and it knows or figures out *how*.

In C++, the objects are structures and the messages are member functions.

You declare each C++ class (user-defined type) in its own header file (with an extension of ".hpp"). A class declaration consists of data elements and declarations of so-called member functions which can be invoked on instances (objects) of the class.

Small functions may be declared in-line in the header file (more about that later).

Unless all the member functions are declared in-line, a class usually has a file of code (with an extension of ".cpp") where all the member functions are defined. This file is compiled into an object module (not to be confused with "objects," which are instances of a class — ugh, what a syntactic mess).

C++ programming at the top level consists of simply including all the header files for the classes you want to use, declaring instances of those classes (objects) just like you would declare integers or floating-point numbers, calling member functions for the objects, and linking in the object modules. I've illustrated this in Figure 1. The main() routine simply manipulates objects.

A Classy Lingo

The main feature which makes C++

By Bruce Eckel
Eisys Consulting
1009 N. 36th Street
Seattle, WA 98103

Figure 1 —Main MicroCad Code
Copyright 1988 Bruce Eckel
Permission required to distribute source

```
// file: microcad.cpp
/* The main driver program for the CAD system.
   This includes all the other types I've defined,
   declares some instances, and makes them dance
   around. */
#include        <stdio.h>
#include        <stdlib.h>
#include        <conio.h>
#include        <msmouse.h>
// flash graphics declarations:
#include        <fg.h>
// (All items starting with "fg" are from
// the flash graphics library)
#include        <string.h>
#include        "msmenu.hpp"   // Figure 2
#include        "circle.hpp"   // Figure 5
#include        "square.hpp"   // Figure 7
#include        "line.hpp"     // Figure 9
#include        "shapelst.hpp" // Figure 11

// associate unique integers with the following:
enum { CIRCLE, SQUARE, LINE, MOVE, DELETE, EXIT};

struct menu_s my_menu[] = {
  " circle",    CIRCLE,
  " square",    SQUARE,
  " line",      LINE,
  " move",      MOVE,
  " delete",    DELETE,
  " exit",      EXIT,
  "", -1             /* end marker */
};


main() {
unsigned x, y;
int quit = 0;  // flag
int result;
shapelist list;
if (fg_init_all () == FG_NULL) {
  fputs ("Unable to open graphics device.\n",
         stderr);
  exit (1);
  }

// create an msmenu and attach our menu:
msmenu mouse(my_menu);

while (!quit) {
    // look for right button press:
    if (msm_getstatus(&x,&y) & 2) {
        // get a menu selection:
        result = mouse.get_selection(x,y);
        // change from mouse to fg coords:
        mouse.translate_coords(&x,&y);
        switch(result) {
            case CIRCLE:
                // create an object on the free
                // store via "new" and add it
                // to the list:
                list.insert(new circle(x,y));
                break;
            case SQUARE:
                list.insert(new square(x,y));
                break;
            case LINE:
                list.insert(new line(x,y));
                break;
            case MOVE:
                mouse.cross_cursor();
                mouse.wait_left_pressed(&x,&y);
                // you can declare a variable at
                // the point of use:
                cadshape * mv = list.nearest(x,y);
                // (I think nearest() needs work).
                // object pointers use arrows
                // for de-referencing members:
                mv->erase(); // pick it up
                mouse.wait_left_released(&x,&y);
                mouse.translate_coords(&x,&y);
                mv->move(x,y); // put it down
                mouse.default_cursor();
                break;
            case DELETE:
                mouse.cross_cursor();
                mouse.wait_left_pressed(&x,&y);
                mouse.translate_coords(&x,&y);
                cadshape * rm = list.nearest(x,y);
                rm->erase();
                list.remove(rm);
                // free the memory created w/ new:
                delete rm;
                mouse.default_cursor();
                break;
            case EXIT:
                quit++;
                break;
            default:
                break;
        }
    }
}
    fg_term(); // back to text mode
}

/***/
```

"Give me one good reason to give up C."

"How about 43?"

Modula-2 saves more time and money than any other programming environment.

1. High-level language
2. Readable, maintainable code
3. Ideal for team programming
4. Supports multi-tasking
5. Emerging international standard
6. Pascal or C programmers learn it in hours
7. Language for modern engineering
8. Consistency checks across modules
9. User control over exported/imported objects
10. Traps most programming errors
11. Fewer bugs in final code
12. Easy low-level access

The LOGITECH Modula-2 programming environment goes far beyond the language.

13. Faster project throughput
14. Corporations rely on it
15. Adds a rich set of tools to the language
16. Best debuggers for any language
17. Configurable, easy-to-use text editor
18. Integrated environment
19. Powerful windowing interface
20. Compiles twice as fast as MS-C
21. Code as fast as the best C compilers
22. Mature and reliable
23. Extended library
24. Standard object format
25. C libraries can be used
26. Supports EGA 43-line mode
27. Automatic MAKE
28. Flexible overlays
29. Price/performance leader

Figure 2 — Square Shape Declaration
Copyright 1988 Bruce Eckel
Permission required to distribute source

```
// file: square.hpp
#ifndef SQUARE_HPP
#define SQUARE_HPP
#include <fg.h>
#include "cadshape.hpp"

class square : public cadshape {
    fg_box_t small_box;
  public:
    void draw();
    square(unsigned x, unsigned y) : () {
        x_center = x; y_center = y;
        draw();
    }
    void erase();
    ~square() { erase(); }
};
#endif SQUARE_HPP

/***/
```

Figure 3 — cadshape Declaration
Copyright 1988 Bruce Eckel
Permission required to distribute source

```
// file: cadshape.hpp
/* A polymorphic base class for all shapes.
   Almost all functions are virtual, so they
   can be redefined in derived classes. Then
   we can make a list of shapes and draw()
   each shape in the list without knowing
   exactly what it is. */
#ifndef CADSHAPE_HPP
#define CADSHAPE_HPP
class cadshape {
  protected:
    unsigned x_center, y_center;
  public:
    // virtual functions must have SOME
    // definition in the base class, even
    // if it's just empty:
    virtual ~cadshape() {}
    virtual void draw() {}
    virtual void erase() {}
    void
    move(unsigned new_x, unsigned new_y ) {
        x_center = new_x;
        y_center = new_y;
        draw(); // call proper virtual function
    }
    unsigned long
    range( unsigned xr, unsigned yr ) {
        // a measure of distance between a
        // selected point and this object's center
        unsigned long xx =
          xr > x_center ?
            xr - x_center : x_center - xr;
            // (ternary if-then-else)
        unsigned long yy =
          yr >  y_center ?
            yr - y_center : y_center - yr;
        xx *= xx;
        yy *= yy;
        // delta x squared + delta y squared:
        return xx + yy;
    }
};
#endif CADSHAPE_HPP

/***/
```

more than just "a better C" is the class. Defining a class means making a new data type which the compiler treats as if it were a built-in type (enforcing the proper syntax rules).

Each class contains hidden information (usually data, but may include functions, which are only available to member functions of the class), and public information (usually member functions, but may include data).

This separation between public and private elements is called "data hiding" and is controlled by the "public" keyword. The public member functions define the interface to the class.

Figure 2 is a small declaration for one of the shapes manipulated by the CAD system. In it you can see some private data (a structure from the Zortech flash graphics package) and some public member functions.

Classes may be inherited from other classes simply by mentioning the name of the base class. In Figure 2, "class square : public cadshape" means square is inherited from cadshape, so it automatically contains all of cadshape's data structure and member functions.

The "public" keyword means that the end user will have access to all of cadshape's public parts (otherwise only square could use them).

In the ".cpp" code files, you'll see member function names with the class name followed by the "::" scope-resolution operator. This tells the compiler that it isn't an ordinary function; it's a member function for the class.

Generally, each class contains a constructor (a function with the same name as the class) and a destructor (the class name with a tilde). The constructor is automatically called by the compiler to perform initialization when an object is declared, and the destructor is automatically called to clean things up when the object goes out of scope.

Virtually Anything ...

There are two reasons to use inheritance. The most obvious is that you don't want to reinvent the wheel — you just want to use most of the features in a class that someone else has already written and tested. This is the reusable code concept.

The second reason for inheritance is to implement polymorphism, which is why I've used it here. Figure 3 shows a declaration for the cadshape class, a base class for all the shapes used in the MicroCad program.

Notice that almost all the functions are defined using the virtual keyword. Any function can be defined in a base class and redefined in an inherited class.

An object of an inherited class can be treated as an object of the base class — if you use a pointer to an inherited class object for calling a function in the base class, everything works fine, except you can only call the base class function this way.

A virtual function, on the other hand, allows you to call the proper function for an inherited class via a pointer of the base class type.

The MicroCad program has a virtual function called draw in the cadshape base class. Cadshape is inherited into circle, square and line, each of which has a different way of drawing itself.

What we want to do is manipulate a list of shapes and thus not have to worry about how to draw them, erase them, etc. Virtual functions allow this — they call the correct draw() function at run-time regardless of whether the shape is a circle, square, line or some new type of shape you've just added.

Virtual Function Operation

Normal C++ member function calls are resolved by the compiler, which knows the specific member function at compile time and simply makes a call to the absolute address of that function. In this case, the size of the object is simply the size of the structure to hold all the data elements you've defined.

If, however, you declare any member function(s) to be virtual, an extra pointer is secretly added to the structure by the compiler (to see this, try taking the sizeof an object with and without virtual functions). The pointer is the address of a table of function pointers — all the addresses of the virtual functions.

When we use virtual functions, the address of the virtual function table is bound to the object AT RUN TIME. Calls to member functions are resolved by dereferencing the virtual function table, also at run time (although proper function call syntax is still enforced by the compiler). This way, you can manipulate pointers to a polymorphic base class; the correct member functions are always called.

To use virtual functions, you *must* pass pointers rather than the objects themselves. You cannot say that an object of a derived class is actually a base class object, but you can say that a derived class pointer is a base class pointer.

This is because a derived object is usually larger than the base class it was

Announcing Modula OS/2. The operating system finally catches up with the language.

30. Support for dual mode operations
31. Dynamic link libraries
32. For standard/extended version of OS/2
33. Multiple threads
34. Virtually unlimited program size
35. Makes mixing languages easy
36. Most powerful editor under OS/2
37. Background compilation while editing
38. Run-time checks
39. Stack checks even in threads
40. OS/2 uses Modula-2 parameter passing mechanism
41. Upgrade available for Modula-2 DOS users
42. Direct Hotline and free Bulletin Board support for all Modula-2 products

43. It's affordable!

Call toll-free:
800-231-7717
In California:
800-552-8885

Please send me:

☐ Modula-2 Compiler Pack (DOS)	$ 99.00
☐ Modula-2 Toolkit (DOS)	$ 169.00
☐ Modula-2 Development System (DOS, includes Compiler and Toolkit)	$ 249.00
☐ Modula OS/2	$ 349.00
☐ Modula-2 VAX/VMS version	$2,500.00
Shipping & Handling (per item)	$ 6.50
CA residents add applicable sales tax	$______
Total	$______

☐ Check/money order included
☐ Visa ☐ MasterCard

Card Number ______ Exp. Date ______
Cardholder Name ______
Authorized Signature ______
Ship to:
Name ______
Address ______
City ______ State ______
Zip ______ Phone ______
Offer valid in U.S. Only Dealer inquiries welcome.
Educational prices available. CL988
Send to:

LOGITECH

Logitech, Inc.
Attn: Coupon Redemption Program
6505 Kaiser Drive, Fremont, CA 94555

In Europe, contact:
LOGITECH SA in Switzerland
Tel: ++41 (0) 21-869-96-56
In the United Kingdom, contact:
LOGITECH UK
Tel: ++44 (0) 525-22-22-11

derived from (you usually add data elements when a class is inherited). You cannot simply treat it as a base class object since the wrong size would be passed around. Pointers to objects are all the same size, so they can be passed around anonymously.

The C syntax of fetching member elements given a structure (structure.data_element) and given a structure pointer (structure_pointer->data_element) extends to member functions (object.function() and object_pointer->function()). It sometimes helps if you remember that an object is just a structure.

In Figure 1, you can see examples of the use of virtual functions in the MOVE and DELETE cases.

Inline

I've used a fair amount of inline code in the examples; this helps reduce the space required to print the program in the magazine. Inline code is placed in the compiler's symbol table, however (except when a function contains flow control — in this case the compiler secretly makes it a non-inline function).

If the compiler runs out of memory, try changing inline functions to ordinary (e.g., static) functions.

Inline code has its advantages, allowing you to (1) access private members of a class, and (2) perform a repetitive group of statements with a more meaningful function name, both without the overhead of a function call.

But you should be aware that an inline function is distinctly different from a normal function definition. No code is generated when the in-line function is defined; the code is inserted directly whenever you call the function.

Parameters aren't pushed and there's no assembly-language "call," which saves time. But the code is duplicated every place the inline function is called. That means your program can take up a lot more space if you write big inlines and call them a lot.

Dynamic Memory Allocation

Ordinarily, the size of an object and the number of objects needed are known at compile time. Sometimes, however, you don't know how big something is (i.e., a matrix) or how many of something you need (i.e., shapes in a cad system) until run time.

When the compiler knows how big or how many, it can automatically place objects on the stack. If size or number can't be known until run time, *you* have to figure it out in the program and allocate

Figure 4 — The shapelist Class
Copyright 1988 Bruce Eckel
Permission required to distribute source

```
// file: shapelst.hpp
/* Two classes to manage a list of cadshapes.
   A shapelist contains a list of (what else)
   shapelist_elements, which simply hold
   a cadshape and a link to the next
   element. The usual linked list stuff:
   you can insert, step through, etc. */
#ifndef SHAPELST_HPP
#define SHAPELST_HPP
#include "cadshape.hpp"

// tell the compiler the class exists:
class shapelist;

class shapelist_element {
    cadshape * shp;
    shapelist_element * next;
  public:
    // allow shapelist access to the
    // private elements of this class:
    friend shapelist;
    shapelist_element(cadshape * s,
      shapelist_element * hd) {
        shp = s;
        next = hd;
    }
};

class shapelist {
    shapelist_element * head, * current;
  public:
    shapelist() { current = head =
       new shapelist_element(
          (cadshape *)0, (shapelist_element *)0);
    } // end of list marked by null pointers
    void insert(cadshape * s) {
        shapelist_element * p =
           new shapelist_element(s, head);
        head = p;
    }
    void reset() { current = head; }
    cadshape * next();
    void remove(cadshape * s);
    // hunt through the list and find the
    // nearest element to x,y:
    cadshape * nearest(unsigned x, unsigned y);
};

#endif SHAPELST_HPP

/***/
```

space on the "free store." This is called "dynamic memory allocation," and is accomplished in plain C via library functions (malloc(), free() and variations on those names).

In C++, dynamic memory allocation was too important to be left to a library function, so it was incorporated into the core of the language via the keywords "new" and "delete" (to free the space allocated with "new"). The "new" keyword is used in Figure 1 in cases CIRCLE, SQUARE and LINE. The "delete" keyword is used in case DELETE.

Linked Lists

The MicroCad program (Figure 1) just manages a linked list of cadshapes. This linked list is yet another class called shapelist, which manipulates cadshape holder objects called shapelist_elements (see Figure 4). There are shapelist member functions to insert a new shape, remove a shape, step through the list, and hunt for the shape nearest a pair of coordinates.

Left As An Exercise ...

I've created the framework for the program. Here are some features you might want to add:

- Create an object to hold coordinates. This (instead of the x,y pair I use) can be passed between member functions to convert from

mouse to flash graphics coordinates and back.
- Stretchable objects, with dashed lines to indicate size.
- Dashed outlines when an object is being moved.
- Create a super-shape which holds any number of other shapes allowing the user to create new composite shapes. Storage and retrieval of these shape libraries.
- A method to translate a shape outline into some independent graphics representation, which can be sent through filters to generate output for PostScript, Epson, etc.
- The mouse menu should be created as a simple graphics block and painted onto the screen (instead of drawing it character by character, as I've done here). This would be much faster.

Building MicroCad

The Micro C RBBS contains the complete MicroCad program (all the code shown here, several other listings, and a makefile). Once you have Zortech C++ installed, simply un-archive the code in its own directory and type "make."

Next Time

In the next issue I'll be looking at *real* CAD systems for designing and laying out printed circuit boards.

Editor's Note: Bruce often refers to articles in previous issues. You can get a book of all his hardware articles, Computer Interfacing with Pascal & C, *plus a disk including the source code and numerous other examples, by sending a check for $30 (plus 8.1% Washington State sales tax) to: Eisys, 1009 N. 36th Street, Seattle, WA 98103. Bruce is also putting together a library of public domain C++ source code. The disks are available directly from Bruce at the above address.*

♦ ♦ ♦

CITIZEN MATE/12 286 SYSTEM

80286 With 12.5 MHz Clock Speed
has on the Mother Board:
ONE Meg RAM with 1 Wait State
Video Controller Supports EEGA, EGA
CGA, MGA, Hercules and
Plantronics Color Plus
Controller Provides Support for
Two Hard Drives and Two Floppy
Drives, 5.25 and 3.5 Capability
Mouse, Parallel and Two Serial Ports

1.2 Meg Floppy Installed
32k Hard Drive Cache Installed
101 Enhanced Keyboard
MS-DOS 3.3 With GWBASIC
Small Footprint
Standard 1MB Expandable to 4MB
Novelle Compatible
Nation Wide Service

********************************$1669.00

XT CLONE SYSTEMS
PLEASE CALL FOR CURRENT PRICE

HARD DRIVES FOR XT AND AT

ST-225 KIT FOR XT (20 MEG)	$ 259.00
ST-238 KIT FOR XT (RLL 30 MEG)	$ 279.00
ST-251 FOR AT (40 MEG)	$ 359.00

MONITORS

Color Monitor RGB (CGA)	$ 255.00
Color Monitor RGB (EGA)	$ 355.00
Monochrome TTL (Green)	$ 95.00
Monochrome TTL (Amber)	$ 105.00
EGA Color Video Card	$ 129.00

CITIZEN PRINTERS

MODEL 120D	120 CPS	9"	$ 179.00
MODEL 180D	180 CPS	9"	$ 199.00
MODEL MSP-15E	160 CPS	15"	$ 359.00
MODEL MSP-40	240 CPS	9"	$ 319.00
MODEL MSP-45	240 CPS	15"	$ 439.00
MODEL MSP-50	300 CPS	9"	$ 419.00
MODEL MSP-55	300 CPS	15"	$ 499.00

CASCADE ELECTRONICS, INC.
ROUTE 1 BOX 8
RANDOLPH, MN 55065
507-645-7997

Please ADD Shipping on all Orders
COD Add $3.00 Credit Cards ADD 5%
MN Add 6% Sales Tax Subject to change

Reader Service Number 15

Hercules Graphics Printer Dump

There sure isn't much privacy at Micro C. Where else could someone assign a programming project and have it announced to the world. (Of course it is in C. "Hello World")

Ah, SOG. I had high hopes for this one. You see, after spending a year with more article ideas than I could shake a stick at, I'd come up with a blank. Nothing. At least nothing I could beat into a useful piece. But certainly SOG would change that. What better place to harvest a few topics for the coming year?

Unfortunately, while vastly stimulating and amusing, SOG left me feeling brain-dead. Still no ideas. No problem though — Unca Dave came to the rescue.

Necessity Is The Mother ...

Lately, Dave's been dragging his behind in a little lower than usual. It's those late night, early morning bouts with **The Program**. (Has something to do with airplanes. I suppose that's why it crashes so much.) So I was more than happy to help out when he said he needed a way to dump a Hercules graphics screen to printer. Now, I'm normally a staunch advocate of the quick and dirty school of programming. But, for Dave, I decided to compromise my morals this time and just be quick.

I had three printers to play with: Gemini-10X, Citizen 120-D, and Panasonic KX P1595. Parallel printers of the clone world live in something approaching harmony. Escape sequences used for programming show an amazing amount of compatibility. Armed with that knowledge, I developed the code on my trusty 10-X, fully expecting it to run with all three printers.

It almost worked out that way.

Printer Setup

Want to print characters? No problem. You throw text to the printer like meat to a dog. The printer chews away for a while and out come the characters on paper. (The dog's output is slightly different, although it does resemble some of the printer output I've generated.) Graphics output doesn't present much more of a challenge.

First off, we need to initialize the printer. prn_setup (see Figure 1) sends out an ESC '@'. This sets the printer to most of its power-on defaults. The carriage return sets up the print head for the first line of output. Otherwise printing could easily start in the middle of a line.

The last line of prn_setup sets line feeds to 8/72". These printers have nine pins in the print head. It makes sense to use just eight pins since the printer can only accept a byte at a time. Each pin prints a 1/72" dot, so 8 pins means ... 8/72".

Grabbing A Screen

The next job involves reading video memory. I'm not going to rehash the confusion of video addressing on the Herc card. Take a look at "A Hercules Primer" in *Micro C* issue #39 for the details of how get_line works. Notice the unary one's compliment operator (~). We need to invert the image since "on" video pixels show up light, but "on" printer elements are dark.

Printing the screen sideways on paper has two advantages: The image fits, and

Figure 1 — Hercules Graphics Screen Dump Functions

```
void prn_out (unsigned char X)
{
  _DX = 0;                      /* specify first parallel printer */
  _AL = X;
  _AH = 0;                                 /* service 0 - byte out */
  geninterrupt (0x17);                      /* printer interrupt */
}  /* prn_out */


void prn_setup ()
{
  prn_out (27); prn_out (109); prn_out (0);/*standard mode, Pan*/
  prn_out (27); prn_out ('@');                   /* reset printer */
  prn_out (13);                                /* carriage return */
  prn_out (27); prn_out ('A'); prn_out (8);      /* set 8/72 LF */
}  /* prn_setup */


void get_line (int line_num, unsigned char buffer [348])
{
  int col;

  for (col=0; col<348; col++)
    buffer [col] = ~peekb (0xb000, 0x2000 * (col % 4) +
                 90 * (col / 4) + line_num); /* get video byte */
}  /* get_line */


void setup_gr_line (int width)
{
  prn_out (27);
  prn_out (75);                                   /* set 60 dpi */
  prn_out (width % 256);   /* these 2 lines set # graphics ... */
  prn_out (width / 256);              /* ... chars to be sent */
}  /* setup_gr_line */
```

By Larry Fogg
Micro C Staff

the coding is simpler. So get_line reads eight columns of the screen to create eight lines on the printer. The terminology gets a little strange here since video rows are printer columns. Think of turning your monitor over on its right side. Now it's easy to see that we should start printing with the bottom row of the first column on the screen.

Editor's note: But don't leave your monitor on its side, that would make the printed image come out upside down.

Printing Graphics

It's time to fire off the line of graphics. setup_gr_line programs the printer to accept a number of graphic characters equal to the parameter, width. We can only give the printer a byte at a time, which means it'll take two parameters to program more than 256 characters. That explains the modulo and integer division in setup_gr_line. Think of these two parameters as a two-digit, base 256 number.

Parallel printers of the clone world live in something approaching harmony. Interestingly, escape sequences used for programming are surprisingly similar.

Once we're done programming and we start sending the actual graphics data, each of the 8-bit characters fires the eight pins in a combination determined by the character's binary value. Each set bit fires a pin, with the most significant bit controlling the top pin. For example, 11110000b activates the top four pins.

Once print_line has printed the contents of buffer (going backwards, remember we want to start printing with the bottom of each video column), it ends with a line feed. Up until now, the printer was just filling its buffer, not printing. The linefeed triggers the actual printing.

I added the print_blank function in order to center the printer output. It simply prints some number of blank characters with no linefeed. The next graphics line starts filling the buffer where the blanks end, and blanks and graphics print as a single line.

One line done, it remains to do the rest. Since each "line" actually contains eight video columns and the Herc has 720 columns, that means we have a total of 720/8, or 90 lines to print. dump_screen does 'em.

So We're Done, Right?

Pretty straightforward stuff. I put it all together and it worked. Sort of. Once in a while a line feed missed the boat. The following line printed off the right margin. Nasty business.

After much cursing and wailing, I reached the conclusion that one of the graphics characters was not printing. So the line feed at the end of that graphics line was seen as the last graphics character. No more line feed, and two graphics lines would fill the buffer before we

Continued from previous page

```
void print_blank (int width)
{
  int i;

  setup_gr_line (width);
  for (i=0; i<width; i++)
    prn_out (0);  /* print blanks and hold print head position */
}  /* print_blank */


void print_line (unsigned char buffer [348])
{
  int i;

  print_blank (60);
  setup_gr_line (348);
  for (i=347; i>=0; i--)     /* print line in buffer, backwards */
    prn_out (buffer [i]);
  prn_out (10);                                    /* line feed */
}  /* print_line */


void dump_screen ()
{
  int i;
  char line_buf [348];         /* holds a column of video bytes */

  prn_setup ();                            /* ready the printer */
  for (i=0; i<90; i++)    /* print each of the 90 video columns */
  {
    get_line (i, line_buf);
    print_line (line_buf);
  }
}  /* dump_screen */

***
```

printed a line.

I printed a line of each character and, lo and behold, it was ^Z (1ah). What gives? The printer doesn't care what byte it receives. No one's doing any filtering. Or are they?

Time to look at prn_out. It just sends a byte to the printer. In its first, semi-successful incarnation, it looked like this —

```
#define PRN_OUT(X) putc(X,stdprn)
```

putc just didn't like ^Z. I don't know what happened. C is a stream oriented language and ^Z can be used to substitute for garbled characters in a transmission. But it shouldn't have reduced the total number of graphics characters sent.

Whatever the reason, putc would not reliably send ^Z to the printer. Using the printer interrupt and Turbo C's handy register variables for prn_out solved the problem.

By the way, if it hadn't been for the fractal I used testing the screen dump code, I probably never would have found this bug. With something as wild as a plot of the Mandelbrot set, you're almost guaranteed to print each of the 256 possible graphics characters. And some of you thought fractals were useless. Shame!

We're Really Done, Right?

Wrong. I sent my newly debugged program home with Dave. It ran perfectly on his Citizen. I hauled it back to try on Cary's Panasonic — and watched it auger into the antistatic rug in flames.

It wouldn't let me control the size of the linefeeds. And, while line feeding too far creates an interesting zebra-like screen dump, it wasn't quite what I had in mind. *Editor's note: does make fractals look better.* Finally, in desperation, I called Panasonic's technical help line.

I do a fair amount of technical support over the phone so it's always fun to be on the other end. After three days of busy signals I finally got a real live Panasonic technoperson. I described my problem and he asked if I'd set the mode DIP switches correctly.

I sent my newly debugged program home with Dave. It ran perfectly on his Citizen. I hauled it back to try on Cary's Panasonic — and watched it auger into the antistatic rug in flames.

Great. I waited three days for some clown to tell me nothing. Of *course* I checked the DIP settings. But I was nice anyway. And, just to humor him, I rechecked the DIP switches. You guessed it, he was right. Never, ever, doubt the word of your tech support person. They always tell the truth and they're never wrong.

Editor's cautionary note: check for the red S on the cape before assuming anything. You can do it from your phone booth.

The Panasonic reads its DIPs on power up, but you can reprogram the mode at any time. Hence the first line of prn_setup. It bumps the Panasonic into the right mode (standard, or Epson compatible), and has no effect on the other two printers.

Other Modes

I chose the low resolution mode for two reasons. First, it's commonly available. Most all of the mainstream printers support it. And you want your programs to run on as many systems as possible.

Second, for a simple screen dump, we've already matched the resolution of the Hercules screen. There's no need to get into the higher resolution printer modes. One of these modes deserves mention, though.

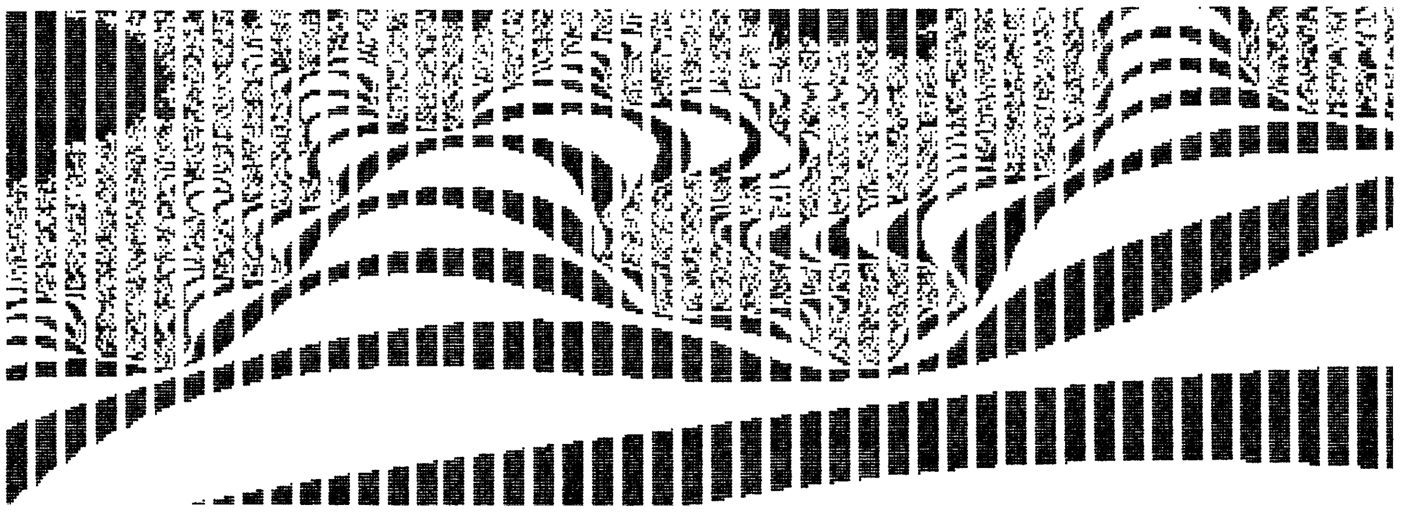
The Panasonic and the Citizen support a 72 dots per inch (dpi) mode, which gives you a one to one aspect ratio *on the printer.* That doesn't mean the print out will more closely resemble the screen. (The standard density, 60 dpi mode is a wee bit squashed. 72 dpi would be flatter still.) It does mean that you can send a square to the printer with 100 dots per side, and it'll be a true square on paper.

We're Done

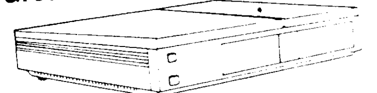
These should be fairly useful and portable (between printers) functions. As usual, I haven't prettied up the code at all. If you want to protect your program from brain damaged users who won't turn on the printer, you'll just have to do it yourself.

Let's see, what else would be fun? Sixteen color EGA dumps? Four color printers aren't uncommon. If we grouped four slightly overlapping dots together to represent each EGA pixel ... Wouldn't be that hard to do. Maybe I'll look into it. Or perhaps I'll just go on vacation. Yeah, that's the ticket.

♦ ♦ ♦



Great for the Experimenter!



Pioneer Laser Disc Player

These units were removed from service for upgrades. They run on 100VAC from supplied 120VAC adapter. While they should be working, some may have minor problems so we must sell them on an "As-Is" basis.

- 1-2 mW He-Ne Laser Tube
- Laser Power Supply
- 2 Front Surface Mirrors
- Two 1/2" Voice Coil Actuated Oscillating Mirrors
- One Beam Splitter
- Two Optical Lenses
- One Optical Detector
- Mini Gear Reduction Motor
- All Controlling Electronics
- Assorted Switches, Fan, Solenoid

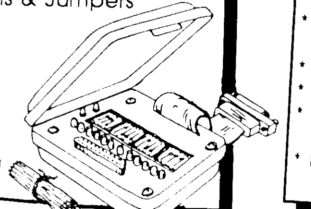
THESE UNITS ARE OEM MODELS AND REQUIRE A COMPUTER CONTROLLER (NOT AVAILABLE FROM HSC). THEY CANNOT BE MANUALLY CONTROLLED!

$99.00

NEW!

RS-232 BREAKOUT BOX

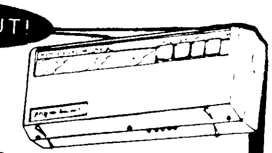
- Switchable Lines
- LED Indicators
- Patch Terminals & Jumpers
- Compact Size



$31.95

CLOSEOUT!

DISPLAY PAGING RECEIVER



- Dual Conversion Superhet 450 MHz Crystal Controlled Receiver Module (Plug-In!)
- Twenty-Char. Alpha-Numeric LED Display (ASCII Encoded, ANSI Char. Set)
- RCA CDP 1802 Based
- Peizoelectric "Beeper" Unit
- Vibrating "Silent" Alert
- Untested - "As-Is"

$9.95 each or two for **$17.95**

IBM / APPLE SOFTWARE BLOWOUT!

IBM TITLES

Visi On Graph
Visi On Calc
Visi On Word
~~VisiCalc~~ SOLD OUT!
VisiWord
VisiSpell
VisiTutor for VisiWord
StretchCalc
VisiSchedule
Applications Manager
Desktop/Plan
VisiFile
FlashCalc
VisiAnswer
VisiTrend/Plot

APPLE TITLES

FlashCalc
VisiCalc
VisiLink
VisiDex
VisiTutor for VisiCalc
VisiCalc (Advanced)
VisiFile

* **VisiCorp Closeouts** *

Priced from $6.95 to $12.95

* Mouse and Basic required for some programs

COMPUTER CHASSIS with POWER SUPPLY

These attractive system chassis were manufactured by Televideo for the TS806/20 Computer System. They are brand new and include the following features:

- Heavy Duty Plastic Case
- 17" x 17" x 8" O.D.
- Hinged Drive Mounting Assembly for 2 Floppy Drives and 1 Internal Hard Drive
- +5 +12 -12 Power Supply
- IEC Receptacle, Power Switch
- Fan

NEW! IN ORIGINAL BOX! **$69.00**

Mitsumi UVEX-AW51P

UHF / VHF VARACTOR TUNER

- 3 1/2" x 2" x 1/2"
- Pinout on Case

TWO FOR **$17.95**

HALF HEIGHT EXTERNAL DRIVE ENCLOSURE

- Attractive Low Profile Case
- 19" x 15" x 3" O.D.
- Fits nicely directly under PC
- Standard IBM Colors
- Bezel fits One 5 1/4" and One 3 1/2" Drive Only

$29.95

SPECIAL!

DELTEC AC LINE CONDITIONER
400 WATT

- Ideal for you PC or Any Equipment
- Eliminates AC Line Noise and Regulates to a Constant 120VAC
- Reliable Ferroresonant Transformer
- Attractive Case with Power Cord, Outlets, and Switch/Circuit Breaker

$149.00

300W, 2500W, 5000W also available

TEAC FD-55B DISK DRIVE

- 5 1/4"
- IBM COMPATIBLE
- 360K
- 90-DAY WARRANTY

$89.95

*ENGINEERING! * TEST! * DEVELOPMENT!*

AT/XT 3-SLOT MOTHERBOARD EXTENDER

- Fused Extender Card
- One 16 Bit Slot
- Two 8 Bit Slots
- Test Points for All Bus Points
- Power Connector
- Cables Included
- Not an Expansion Chassis!

$89.95

WE SHIP C.O.D.!

Call the HALTED ELECTRONIC RESOURCE BBS! (408) 732-2814

HSC of Santa Clara
3500 Ryder Street
Santa Clara, CA 95051

VISA

AMERICAN EXPRESS

HSC of Sacramento
5549 Hemlock St.
Sacramento, CA 94928
(916) 338-2545

HSC of Santa Rosa
6819 Redwood Drive
Cotati, CA 95841
(707) 792-2277

HSC Since '63!

Call Now!

Outside California ***800-4-HALTED***

California Residents ***408-732-1573***

TERMS: Minimum order $10. California Residents add 7% sales tax. Prepaid orders sent freight C.O.D. or call for charges. Shipping will be added to credit card and C.O.D. orders. $2 Handling charge on orders less than $25. Prepaid orders over $100 use money order or certified check. Please no not send cash. Some items limited to stock on hand. Prices subject to change.

Reader Service Number 11

The Peripheral Technology PT68K-2

A Hacker's 68000 System

Usually I look a bit askance at articles that are this effusive. I have yet to meet a prince who doesn't have a few (remaining) warts. However, Jack's enthusiasm for this board reminds me of my early feelings toward the Big Board. And, as you'll see, he's very aware of the warts.

I've got a new toy, and I *love* it! It's the PT68K-2 from Peripheral Technology, and it's the computer I've been searching for for years. It's cheap, powerful, fast, and fun to use. It's the perfect hobbyist's system. Perhaps more important, it's *here*, and it's *real*.

For the skim-readers among you, the PT68K-2 is a new personal computer, targeted to the hobbyist, and built around Motorola's 68000. The essential details are given in the Figure 1.

As you can see, it's a serious computer, with the performance of a hot AT clone. But statistics alone don't explain why I'm so excited. The reason requires a little history.

A Little History

I'm a computer professional. For more years than I care to think about, I've been getting paid to work with computers. Sometimes it's fun, sometimes it's not. When I'm at work, I write software the way my bosses want me to, in the languages they want me to, to solve the problems they want solved.

I don't have much to say about the operating systems or the tools, so I'm forced to be a closet hobbyist. When I'm at home, I take a busman's holiday. That way I can use the computer and software of my choice to do the things I want to do.

In that sense, I probably have a lot in common with the hot rodder or the radio amateur. Many of you are also hard core hobbyists, or you wouldn't be reading *Micro C*.

The Ideal Hobby Computer

I wanted performance. That means a modern, fast 16 or 32-bit CPU. Lots of programmers prefer the 68000 architecture. Its instruction set is simple and rational, a logical successor to the Z80.

I wanted lots of fast RAM, serial and parallel ports, floppy and hard drives, clocks, memory mapped video ...

And, naturally, the hardware should be hackable, which means an expansion bus. (Too bad I couldn't use the IBM bus with all the incredibly cheap I/O.)

It must be cheap and, perhaps, buildable in modules.

What about the software? Nothing fancy here ... the operating system (OS) and tools should be simple enough for one human to understand and customize in one lifetime. The software, too, must be cheap.

Above all, the hardware and software should be *rational*. There's no point inviting frustration by having to deal with irrational designs. (That leaves out most of the more popular computers!)

Now, since this is a hobby computer, better put a debug monitor in ROM. And there should be some consideration for reliability and testing built in.

The PT68K-2

There are a number of reasonably priced 68000-based computers. The Mustang-08 and -020 from Data-Comp, the QT series from Frank Hogg Labs, and the Marion Systems MS68K come to mind. Closer to home, *Micro C* has supported efforts to build a 68K-based public domain system and has run several articles on Joe Bartel's Tiny Giant from Hawthorne Technology.

However, my list of needs describes the PT68K-2. Frederic Brown of Peripheral Technology must have been reading my mind when he designed this board.

Like it or not, the PC phenomenon has given us cheap hardware. The prices of such things as keyboards, enclosures, and power supplies represent incredible bargains of price/performance. That's one thing that's kept me away from the other 68000 systems. Even though their mainboard may be cheap, the price for the whole system (power supply, terminal, etc.) is pretty steep.

The PT68K-2 is designed around the PC, right down to the cable connectors. Plus it's got a PC bus, which accepts any PC-compatible I/O boards.

The standard OS is SK*DOS, from Peter Stark of Star-K Software Systems. For a one-man and relatively new OS, SK*DOS is surprisingly mature, and quite bulletproof.

That's because it's a clone of Flex, a popular 6809 OS, and Peter has been working with Flex for many years. To a Flex user, SK*DOS will seem like an old friend. To the rest of us ...

Peter also supplies HUMBUG, the ROM monitor, and utility software.

It's my understanding that the PT68K-2 is the result of a collaboration between Peter and Frederic, and I think this shows in the integration of the package.

Putting It Together

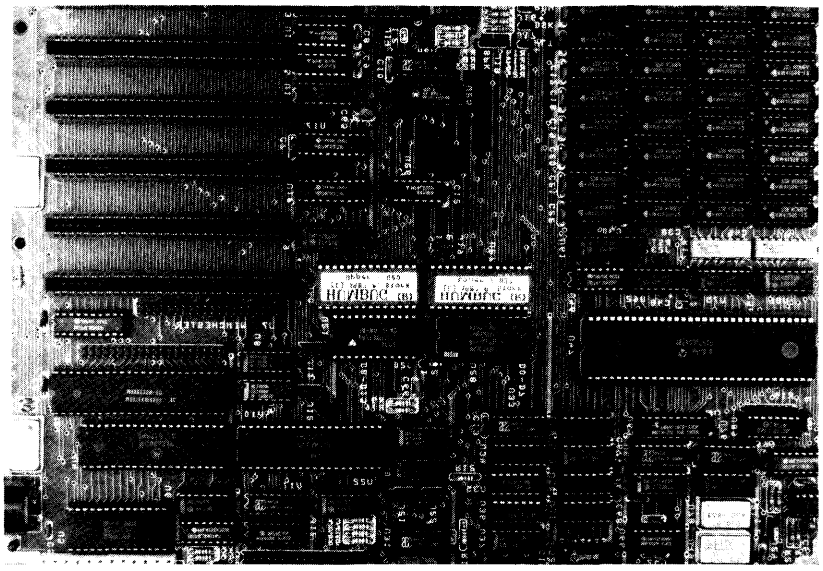
The PT68K-2 motherboard holds 1MB of RAM, so no RAM board is needed (or accepted ... the bus is only good for I/O). Also on the motherboard are the 4K static RAM, four (count 'em) serial ports, two parallel ports (one of which is Centronics compatible), and a floppy disk controller.

The minimum kit is $200. You don't get much of a computer for this — no dynamic RAM, and no floppy controller. But it's a computer.

Add a terminal and power supply, and you're running with HUMBUG in ROM and 4K of static RAM. The ROM even includes a BASIC interpreter. It's got to be one of the cheapest 68K systems around!

Add another $200 for 512K of

By Jack W. Crenshaw, Ph.D.
1220 E. Idlewild Ave.
Tampa, FL 33604



A PT68K-2 Motherboard.

dynamic RAM and a floppy controller, or get a complete kit including 10 MHz CPU for $575. The same motherboard, assembled and tested with 1MB of 10 MHz, zero-wait-state RAM, and SK*DOS goes for $849.

But the PT68K doesn't end with the motherboard. The PC bus gives you easy expandability. Instead of hooking up a terminal, you can plug in a PC-compatible video card ($50 from Peripheral Technology), a monitor, and a PC keyboard.

Finally, if you want a really serious system, add a standard Winchester controller card and hard disk.

I opted for the full-house system, which cost me a total of $1,422, including enclosure, power supply, AT-style keyboard, CRT, and two floppies. (Prices have since gone up.) I pirated a 20MB hard disk from my Kaypro.

To anyone who has ever assembled a PC clone, putting the PT68K-2 together is old hat. Even if you've never assembled another system, putting the PT68K together is a piece of cake. Everything just bolts into place and the cables snap together. In an hour you have a system.

Firing It Up

I fired it up with just the CRT controller and CRT to verify that I had a prompt.

Next I added the keyboard so I could run HUMBUG. Everything looked fine, so I hooked up the floppy drives (just ran a cable from the motherboard to the drives) and booted SK*DOS. Simple.

Adding the hard drive was just as easy. I just typed HDFORMAT, followed the menu prompts, and bingo, it ran perfectly.

It's been that way on everything I've tried. It's so boring ... nothing to work on! It's bulletproof.

For the record, I also ordered an AT clone from a very reputable company at the same time (around Christmas) as the PT68K-2. The PT68K-2 arrived within ten days, and was running in two more. It took me three months to get the AT together and running, plus additional time and phone calls to get it set up.

Design For Testability

Though I bought mine assembled and tested rather than in kit form, I really appreciate diagnostics.

My old Altair taught me that a multimeter isn't enough! You can equip a laboratory with all the test equipment needed to debug a wayward computer.

It's apparent that Frederic and Peter remember those days, too. One of the beauties of their design is the progressive way you put the kit together.

The whole idea of the static RAM and a ROM debugger is that you don't need much to start testing. But they've taken testing to a much lower level. (Peter's most elaborate piece of test equipment is an LED, borrowed temporarily from the front panel.)

I won't detail the whole process here, but I'll give you the general idea. Since the LED is used for testing, it goes in first. Then it's used to test the processor support circuits such as the system clock. The idea is to get the CPU running as soon as possible so it can take over the testing.

I figured that you'd have to have some pretty elaborate software in ROM to handle the tests. Not always. Peter uses a trick that requires *no* software at all! He just straps the data lines to ground so the processor sees a constant 0000.

After the CPU executes 0000, it increments the program counter (to, of course, fetch the 0000 from the next address). Thus, all the address lines get toggled.

By checking the address lines with the LED, you can verify that the CPU is working. (The address lines become square waves which you can use to test the rest of the circuit.)

Once the CPU is running, you install and check the memory decode logic. The ROM and static RAM go in next, and at this point you have a computer.

The whole process is incredibly slick, and I'm kinda sorry I ordered my system assembled. I really missed a lot of fun.

Incidentally, there's a neat little aside to this story. The literature on the PT68K-2 describes a real-time clock, but I found neither clock chip nor battery.

After the system was up and running, I came across a command in SK*DOS called "TIME." I figured, what the heck. I almost fell out of my chair when the PT68K-2 calmly reported the correct date and time. Not only was there a clock, but it already knew the time.

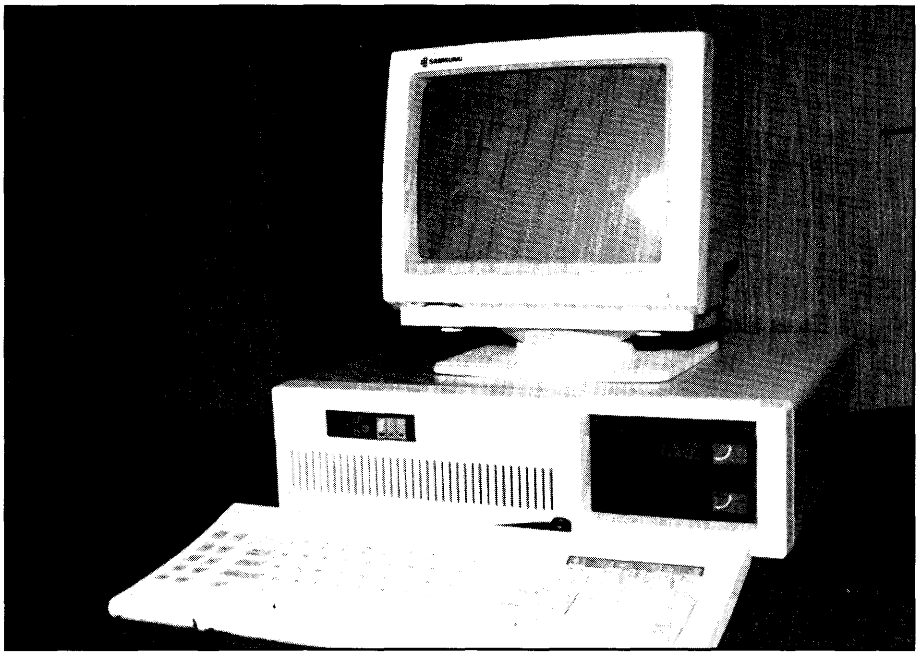
It turns out that Frederic uses a single-chip clock with a built-in battery. The clock has the same pinout as the 6116 static RAM chips (and the clock chip contains RAM, too). You just pull one of the 6116s and replace it with the clock. The software spots the clock and you've got time.

Even better, when the clock chip is in place, the 2K of RAM on that chip is battery-backed-up so HUMBUG uses that RAM to store system configuration info.

My Hardware Problem

Now for my one hardware problem. My copy of MicroEMACS would occasionally go to lunch, and at first I assumed it was a software problem.

However, it turned out that they'd put a 12.5 MHz clock crystal in my 10 MHz system. Fortunately, they put two crystals on the motherboard so a single jumper change slowed me down to 8 MHz. I'll go in and change the 12.5 to 10 MHz when I get around to it, but for now I'm doing fine at 8.


PT68K-2

The Software

A computer without software is about as useful as a car without wheels. What about the software?

SK*DOS reminds me of CP/M. It's modular, consisting of a BIOS, command processor, and file system. There is essentially no part or feature that can't be configured, modified, or replaced (anyone for ZCPR68?). It's also small, about 20K. A hackable size.

Despite its size, it supports custom device drivers, pipes, I/O redirection, BAT files, a RAMdisk, and TSRs. Everything except multitasking. (Peter is working on that.)

You get 40 utilities including: assembler, file manager, the ubiquitous EDLIN, and a BASIC interpreter.

SK*DOS also provides more assembly language help than we ever got from DRI. There are no less than 60 DOS calls, organized as "A-line" software interrupts. These include calls to input and output text strings or decimal and hex numbers.

That's on top of the 17 ROM calls. And you get an INCLUDE file of system equates to make assembly easier.

Although I didn't know 68K mnemonics when I started, I had no trouble recoding some of my CP/M library routines. I found, however, that many of my prized subroutines weren't needed with SK*DOS, their functions were available as system calls.

Two Versions Of C

For $10 Peripheral Technology will send you a sampling of the SK*DOS User's Group library including: a Small C compiler, three (!) editors, an nroff clone, and many other goodies.

Also, Sidney Thompson, president of the User's Group, and Bud Pass are just finishing a full K & R version of C. (I have a beta test copy.) Sidney also sent me the copy of MicroEMACS, developed using the new C.

Finally, I'm starting to see software from other sources. I know of at least two more editors, two disassemblers, and a host of other goodies.

Let's Be Real

I wouldn't be completely honest with you if I didn't tell you that there are some things I don't like.

For those of you who have experience with Flex, SK*DOS will seem like old home week. For the rest of us it's strange. Instead of drives A: and B:, they are 0 and 1. The command processor doesn't understand the familiar "*" and "?" characters in wild cards. Instead you just give the first letters of the filename. Unfortunately, this means typing:

```
DIR .C
```

which not only gives you all the .C files, but also all the .COM files, and any other files whose extension begins with 'C'.

The SK*DOS tools also give me more help than I care for (a carryover from Flex, I think.) When you copy a file over an existing file, SK*DOS stops you with a

6800

and 6800/2/8/9

SOFTWARE

SK*DOS - a powerful DOS for the 6809 ($75) or the 68000 ($140, incl. an edit[illegible] assembler, Basi[illegible] code for [illegible] HU[illegible]

Please see the PT68K-2 review on page 42 of this issue.

[illegible]STEM SOFT-[illegible] including assemblers, text formatters, editors, spell checkers, languages, etc., all very reasonable.

HARDWARE

A wide selection of single-board computers and systems, from $275.

COMBINATIONS

Package deals of fast and powerful computer plus DOS and more, from $350.

STAR-K SOFTWARE SYSTEMS CORP.

BOX 209 • MT. KISCO, NY 10549
914/241-0287

Reader Service Number 40

Figure 1 — PT68K-2 Vital Statistics

```
Mfr.        :    Peripheral Technology, Inc.
                 1480 Terrell Mill Rd., Suite 870
                 Marietta, GA 30067
                 (404) 984-0742

User's Group     SK*DOS Users Group
                 Sidney Thompson
                 181 Greenbriar Ct.
                 Conyers, GA 30208
                 (404) 922-3097 (Eves,Voice)
                 (914) 241-3307 (RBBS,PCBoard)

CPU         :    Motorola MC68000
Clock       :    8 MHz standard
                 10, 12.5, or 16 MHz optional
RAM         :    512-1024K dynamic, 0 wait state
                 4K static, battery backup
ROM         :    32-128K EPROM
Serial      :    Four RS-232 ports
Parallel    :    Two 8-bit ports (one Centronics-compatible)
Time        :    Real-time clock/calendar, battery backup
Expansion   :    6 IBM PC/XT-compatible I/O slots
Floppies    :    Built-in floppy controller, four drive
                 capability
Hard Disk   :    Uses PC-compatible Winchester controller
Console     :    Serial terminal OR PC-compatible keyboard,
                 mono or color monitor
Power       :    Plug-compatible with PC power supplies
OS          :    SK*DOS (free with kit) or OS-9
Software    :    Debugger in ROM. Assembler, editor, file mgt.
                 tools, disk formatter, supplied.
Other S/W
Available   :    User's Group provides Small C, editors, modem
                 program and many other utilities. Software
                 from vendors includes K & R C, RBASIC & other
                 compilers, EMACS & other editors,
                 disassemblers.
Price       :    Partial Kit (without dynamic RAM, floppy
                 controller, DUART, or clock)                      $200

                 Full Kit       SK*DOS, 10 MHz, 512K RAM 2 serial  $575

                 Assembled      SK*DOS, 10 MHz, 1Meg RAM 4 serial  $849
                                SK*DOS, 12 MHz, 1Meg RAM 4 serial  $899


/***/
```

"File exists … Delete?" message. When you delete a file, it asks you, not once but twice, if you're sure you know what you're doing.

More important, the tools are not as mature as the OS. Peter's trying to provide as many tools as possible — and they work, but many are pretty short on features.

HUMBUG is no competition for DDT and I don't like its command syntax. Neither the Cs nor the assembler support separate linking. But that's because there's no linker. (I've promised to write one as soon as I finish this article, so I can't blame that one on anyone else!)

Peter has developed a truly elegant system of I/O control, suitable for device-independent control of a wide variety of devices, but none of the standard I/O drivers use the system.

While SK*DOS fully supports the PC-type CRT and keyboard, it only does so by treating them in TTY fashion. The standard SK*DOS driver can't read all those neat cursor, keypad or function keys. This isn't a problem because the User's Group has a driver which supports all these.

Disk I/O on the hard disk is quite satisfactory, but floppy writes are interminable. That's because the file I/O is unbuffered, and SK*DOS does a verify after every 256-byte sector. You can turn off the verify flag, but you take a chance that a disk error will corrupt the directory.

SK*DOS also has a flat file structure, which means that when you do a DIR on the hard disk, you get *lots* of files. Now that's a problem, particularly because of the lack of wild cards. Fortunately, a combination of four disk partitions and a copy of a UNIX-like DIR program from the User's Group have made life at least bearable.

Finally, although the PT68K-2 hardware is designed to be fully interrupt-driven, SK*DOS uses polled I/O. No typeahead. Well, that's okay … I've lived without typeahead for five years with my Kaypro, but I was hoping …

However, I *like* SK*DOS. The problems are teething pains rather than something fundamental. I remember how slow the disk I/O was on my Kaypro before I installed Micro C's ROM.

The main thing is, the OS is basically sound and quite bulletproof — a perfectly capable base upon which to build. None of the problems I've mentioned are things that can't be fixed.

And help is on the way. Peter is already working on a new version, which will have faster disk writes, a hierarchical file structure, and typeahead.

Right now, SK*DOS is like CP/M before the public domain guys got hold of it. It's badly in need of a cadre of users to write software for it. There are already a few, including the very active user's group.

If you are not the adventurous sort, you don't have to get SK*DOS with the PT68K-2. OS9 is also available. For the record, I have heard more than one person say that OS9 is the best OS ever written for micros, bar none. Some say it's the best OS, period. But it's $500, the BASIC compiler is another $500. To me, that kind of money runs counter to the whole spirit of the system.

Finally

The system is fast, powerful, modular, and cheap. The hardware and the software are solid. The software tools are not as powerful and/or friendly as, say, Turbo Pascal, but they are adequate. Certainly adequate for creating more and better tools.

If the public domain underground ever gets hold of this computer, there's no telling where it will end.

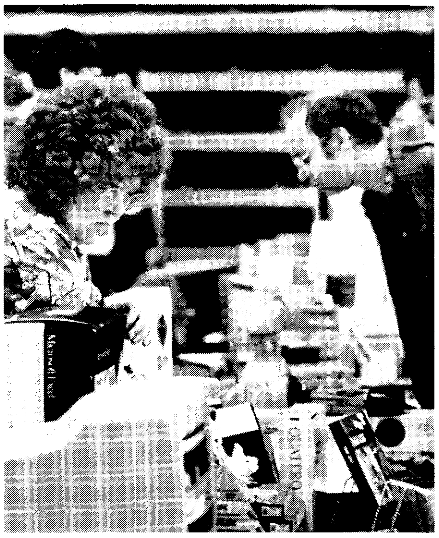
♦ ♦ ♦

Sharing At SOG

This is the kind of SOG piece that I can't write. It's not a disinterested look (I don't think it's possible to attend SOG and remain disinterested), but Barbara definitely has her own perspective. This is also a reprint. It first ran in Oregon Computer News. *Many thanks to its editor for permission to run this piece.*

Approximately 400 "computer techies" and myself gathered in Bend, Oregon, to attend SOG, the Semi-Official Get-together that each year celebrates the publication of *Micro Cornucopia.*

Micro Cornucopia is a bi-monthly computer magazine for the dedicated computer enthusiast. It's interesting to read,


Don and Marilyn Thompson check stock at Microsphere's SOG table.

but also very technical. It informally showcases new languages and operating environments, hardware design such as graphics processors and coprocessors, and mathematical solutions. Leading edge work and experiences are shared with others who may be battling the same problems. *Micro C,* as the magazine is fondly referred to, is all about sharing.

SOG begins with an optional white water raft trip on the Deschutes river. This is the perfect ice breaker. It's very difficult to be stodgy as a dribble of cold water slides down your back, especially when aimed by an overzealous bailer.

This year our sixsome was given the option of taking a series of rapids

Heading for lunch.

without the raft. Five of us accepted and were dropped off on the river bank. As we walked back to the appropriate water entry point, we had time to contemplate our folly. It was a very long walk, too long to gracefully retrace our steps. (Besides, those red-blooded males weren't going to let a skinny female be the first and only taker of this exquisite dare.) The raft trip sets the tempo, and the informal mood continues.

We discarded suits and ties in favor of grungies. Many wear SOG T-shirts from former get-togethers.

Much as early designers of hardware and software pulled up a chair, leaned back and explored problems and possibilities, this year's group also shared. People whose names are synonymous with the computer industry, like George Morrow and Jim Warren, come back and touch roots at SOG.

The Sessions

Two days of intensive sessions began Friday morning at 8 a.m. Software programmers were immersed in compilers, tools, operating systems and debugging techniques. Software designers listened and questioned as the logic of the new modular language Trilogy was introduced. A designer ex-

If that's a 334, then this must be a . . .

plained the processes and problems encountered in developing the Tele operating systems.

PC and drive diagnostics provided solutions to hardware problems. Hawthorne Technology presented its 68000-based single board computer and operating system. Hardware designers examined the pitfalls of obtaining FCC approval for their boards and computer systems. We were all treated to some magnificent color 3-dimensional fractals

By Barbara M. Hall
ProLogic, Inc.
9900 S.W. Wilshire, Ste. 120
Portland, OR 97225

... and the mathematical process that produced them.

Industry leaders PC Tech, Microsoft, Logitech, Intel, Zortech, and new companies who could become familiar names provided this year's speakers. SOGers usually had to choose between two lectures scheduled in the same slot.

Talks such as "Neural Control and Parallel Programming with Transputers," "Ray Tracing on the 34010," and "Numeric Applications" vied for attention. Even with such esoteric titles, speakers often found themselves face to face with an audience which seemed to know as much as they did and was not always of the same opinion. There were lively debates and some of the talks developed into full-blown forums.

In contrast to the technical and often specifically focused nature of many of the speakers' topics, the organizing committee thoughtfully included a couple of business know-how topics. Even technical designers need to know the pitfalls of publishing their books or energizing their employees.

Clark Calkins describes his source code generator.

Relaxed Sharing

This relaxed group feeling of sharing was an important part of the two-day seminar. Lunch was not really break time, but an opportunity to keep the discussion going or greet old friends from a previous year. I heard "remember when we talked about ..." as SOGers updated colleagues on their current progress.

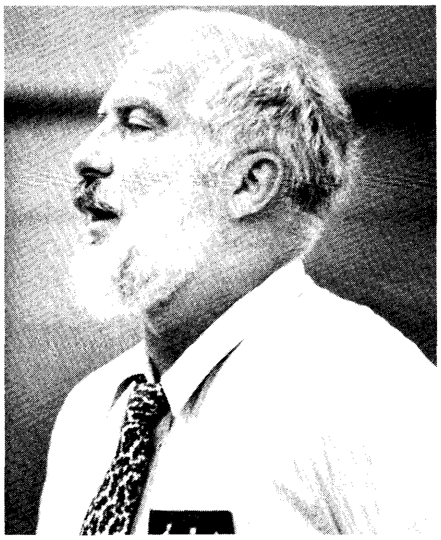
The discussions really take on the air of intensity as small groups gather after the structured day sessions end to talk long into the night over coffee or beer. They explore solutions, kick ideas about, and renew creative energies. It's not all talk ... members of one small group emerged at 4 a.m. with computers they each had built!

History

The Semi-Official Get-together is Oregon grown and bred. Seven years ago 60 people, some from as far away as Australia and Holland, gathered together in Portland for a single day of talk. There was no formal agenda; they shared ideas, potluck and good will. From these modest beginnings SOG was born.

The next year *Micro Cornucopia's* editor Dave Thompson moved to Bend, Oregon, and so did the Semi-Official Get-together. This second annual celebration was attended by 125 *Micro C* enthusiasts who squeezed into Dave's home/magazine space. SOG was growing rapidly and even the Fish Hatchery building in Shevlin Park proved almost too small for the 250 who attended SOG III.

Since those early days, SOG has found a home at Central Oregon Community College. The attendees number from 400 to 450, which is just about the right size for the College facilities.


Jim Warren

The Semi-Official Get-together provides an opportunity for innovators to touch base with others who are in the same league, to work and to play together. It's a network for people from all over. The glitz of the formal computer magazines and commerciality that has invaded the big computer shows have not touched this homespun world. SOG is truly a unique sharing experience.

♦ ♦ ♦

By David Thompson
Staff

Bits From Your Past

You thought you could hide behind your mailbox. You thought you could just sit back and chuckle over other people's misadventures. Wrong, Bunky. This is Micro C, and you're the target of our next exposé.

The evening discussion (at SOG VII) was drifting along pretty much undirected when someone popped up with a question to the fellow sitting next to me.

"How'd you get into computers?"

Well, that led to some pretty interesting stories, and it sure brought memories for me. My first computer was a stock KIM-1. I bought it from an engineer at Tektronix and, along with the board and power supply, he gave me a stack of newsletters called the KIM-1 User Notes. I was impressed. Page after page of object code listings that I could key in on the hex keypad. Wow! I remember pecking at that keyboard for hours to turn that tiny computer into a digital clock. It worked, and it was accurate to within five minutes a day.

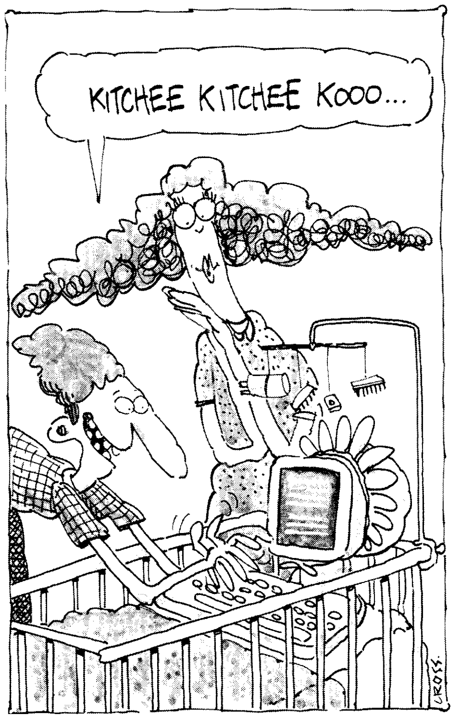
I wrote my own programs, spending hours scribbling out 6502 object code, then entering and testing it. About six years ago I loaned the KIM to an aspiring young computer student. Unfortunately, he and my beloved KIM have disappeared, apparently forever. SIGH!

How did I feel about micros then?

That KIM was unadulterated magic, though through the hex keypad and 7-segment display I only got a glimpse of what was to come. (Now we have real assemblers, interpreters, compilers, text editors, graphics displays, real control of real events, real information processing ...)

How have my feelings changed?

Wow! It's hard to avoid getting lost in the business, the financial, the political, and personalities part of this industry and forget about the computers. We're no longer glimpsing the power, we've got it. We've got the tools. We've got the hardware. Go back five years. Who would have imagined desktop publishing or Turbo C or CodeView?



But I haven't answered the question.

I'm still fascinated by micros. Still excited. It still feels like magic when I hit the switch and hear the hard drive spool up. If anything, I'm even more excited than I was eight years ago.

Now It's Your Turn

Not only did SOGers discover that they were a varied bunch, but they so enjoyed sharing their variedness (my word) that they recommended I do a varied article. (I'd assumed my articles were quite varied enough already.)

After reading through the following questions, I'm sure you'll be as interested as I am to see everyone else's answers. Well, they're just as interested in seeing yours. So grab a cup of tea, take keyboard in hand, and muse long enough to get the memories flowing.

It doesn't have to be wondrous prose, just jot down a page or two or three as you would in a letter to a friend. (If you'd like to be anonymous, say so.)

I'll compile and link the stories and run the results. That way we'll all have a better understanding of this crazy phenomenon that brought us together.

Questions:

(1) What's your life story? (Don't panic, just the good stuff.) Why and how did you get into computers? How large a part are they of your working, home, and/or social life?

(2) What are you doing now? (3 demerits for "I'm reading this form.") What do you plan to be doing in 1 year? 5 years?

(3) Describe your first system. What kind was it? How and why did you get it? How did you feel about it? Where is it now?

(4) How have your feelings about this field changed over the last X years? Why? (Boy, did this question keep a lot of people up all night at SOG. This is a fun one.)

(5) Given complete freedom to choose your work and where you'd do it (leave out the "with whom"), what would it be and where would you do it? (This doesn't have to be computer-related.)

(6) If you were to receive a large box containing a technical item, what would you hope was in the box? Why? (This one's wide open.)

(7) What question would you ask the entire Micro C audience? How would you answer it?

♦ ♦ ♦

TM

EDITING SYSTEM

Solidity, Speed, and Power.

ED is an object-oriented open architecture system for editing programs and manuscripts and manipulating data files. Flexibility, extensibility, and programmability are realized by providing access to the objects which ED itself manipulates. All aspects of ED's appearance and operation can be controlled by the programmer. ED, a restricted or enhanced form of ED, or any of ED's objects, can be embedded in the programmer's applications with no royalties. Objects such as dynamic arrays, windows, data entry windows, macros, menus, browsers, popup directories, and regular expressions are manipulated through normal C function calls. Functions for creating manuscripts are part of ED's design. Stream blocks, word-wrap, tabs, intelligent paragraph formatting, block justification, and pagination behave correctly and execute instantly. Finally, ED can be used to create sophisticated sorts and filters. Macros, search and replace, column blocks, and block sorts allow records to be sorted and selectively removed, and fields to be added, rearranged, and removed.

ED • NUMBER OF BUFFERS LIMITED ONLY BY AVAILABLE RAM • NUMBER OF WINDOWS LIMITED ONLY BY SCREEN SIZE • **POPUP DIRECTORY FACILITY** • COPY • RENAME • DELETE • BROWSE • FILE STATUS • EDIT • EXECUTE • SORT • **SEARCH AND REPLACE** • FORWARD/BACKWARD • CASE SENSITIVE/INSESITIVE • FULLY PARENTHESIZED REGULAR EXPRESSIONS • INCREMENTAL/GLOBAL REPLACE ACROSS ALL BUFFERS • **BLOCK COMMANDS** • COLUMN/LINE/STREAM BLOCKS • SAVE • CUT • DELETE • BLANK • SEARCH-FOR • FORMAT • JUSTIFY LEFT/RIGHT/CENTER • UPPER/LOWER CASE • REMOVE • OVERLAY • REACTIVATE • TAB • DRAG • SORTS • **MACROS** • MENU DRIVEN • VIEW • AUTO-EXECUTE • TIME-DELAY • NESTED • RECURSIVE • SIZE LIMITED BY AVAILABLE MEMORY • **MORE FUNCTIONS** • VIEW/MANIPULATE BUFFERS THROUGH MENU OR KEYSTROKE • INTELLIGENT DIAGRAM/BOX DRAW • ADJUSTABLE SCROLL VALUE • TRANSPOSE LINES/CHARACTERS/WORDS • MOVE-TO-POSITION STACK • VISUAL TAB CREATION • COLUMNAR/ENTIRE-LINE TABS • TIME AND DATE STAMP • RESTORE TYPED-OVER CHARACTERS • CHANGE DIRECTORY • BRACKET/BRACE/PARENTHESIS MATCHING • ENTER GRAPHICS CHARACTERS • EXCUTE DOS COMMANDS • POPUP ASCII TABLE • USER DEFINED MENUS/POPUP FILES • KEYBOARD CONFIGURATION • PRINT BUFFER WITH PAGE NUMBERS, ADJUSTABLE SPACING AND MARGINS • INTERACTIVE TUTORIAL • ALL DOCUMENTATION ON-LINE • WRITTEN IN C AND ASSEMBLER• **EXTERNAL UTILITIES** • INTELLIGENT CHANGE DIRECTORY COMMAND • POPUP DIRECTORY FACILITY • STRING TRANSLATOR • **HARDWARE REQUIREMENTS** • DOS • PC/AT, PS/2, 386 • CGA, MDA, EGA, HERCULES, WYSE 700/AMDEK 1280 • RUNS IN ALL VIDEO MODES, NO FLAGS, NO DRIVERS • 256K • **FOR LITERATURE, A COMPLETE LIST OF FUNCTIONS, AND A DEMO DISK, CALL 201-450-4545.**

VERSION 1.0

PRICE: $265.00

THE AMERICAN
COSMOTRON 80 HOLMES ST • PO BOX 128 • BELLEVILLE, NJ 07109

VISA® MasterCard

Reader Service Number 107

C'ING CLEARLY

Feeling SOGgy

By Scott Robert Ladd
P.O. Box 61425
Denver, CO 80206
(303) 322-7294

Scott was planning a C debugger review this issue. However, Borland's debugger isn't out yet so he's been reduced to SOGging it and futzing with compilers. Here are his latest recommendations on the compiler front.

Squish, squash, squish, squash ... sorry about the noise, but my clothes are still a bit SOGgy. Rule #4316: Don't wear blue jeans on a raft ride if you don't have a dry pair on the bus!

I enjoyed SOG VII immensely! There were great people to talk to, interesting programs to attend, and plenty of white water to go around. One of the most interesting experiences in my life was riding down a river in a rubber boat with a bunch of hackers; we'd yack a bit, then paddle through some rapids, and then pick up the conversation almost in mid-sentence! Wonderful!

This column is being written in *very* early August, just a few weeks after returning from SOG VII, which means you'll be reading it in early October. I came home to find my fish in *really murky* water (yuck) and plenty of boxes on my front porch. (My neighbors like to speculate on what I do for a living. I get several package deliveries a week. I don't "go to work." And they hear me typing and cussing in the wee hours.)

Where Are The Debuggers?

I *was* going to be talking about C debuggers and debugging technology this time around. However, the centerpiece of my discussion was to be a comparison of Microsoft's CodeView and Borland's new debugger. Unfortunately, the new Turbo C has been delayed! Since lots of people are interested in the Borland debugger, I'll wait until next issue.

Meanwhile, let's dig out the old jackknife and open a few boxes ...

WATCOM C Version 6.5

I really like the people at WATCOM; they're friendly, helpful, and *fun* to talk to. They also put out an excellent compiler, which they've just upgraded.

Version 6.5 of WATCOM C has a number of improvements and additions, mostly to keep up with their competitors.

For instance, every C compiler released these days includes a graphics library. WATCOM C 6.5 is no exception. The new graphics functions are similar in form and operation to Microsoft's. However, the WATCOM library does support the Hercules monochrome card. Microsoft supports monochrome graphics, but only if you use a special TSR.

WATCOM has added a "compile-and-link" control program, which is standard with most other compilers. Interrupt Service Routines (ISRs) can now be created by using the qualifier INTERRUPT in a function definition. A new memory model (Tiny) has been added which allows the creation of .COM programs.

A dozen or so miscellaneous functions, based on requests from WATCOM C users, now disable/enable interrupts, parse file names, and rotate integers. *(Editor's note: Be great for reversing all the numbers on a mailing label.)*

I haven't had a chance to run it through the benchmark suite yet, but WATCOM claims to have improved both compile speed and code generation, and they've included more inline functions. Overall, WATCOM C continues to be a good value, and a competitor in technical quality to Microsoft.

C-Ware Desmet C Version 3.1

Just when many people thought that Desmet C was in its twilight years, they come out with a new version. Desmet 3.1 has a larger library and at least some compatibility with the emerging (lordy, how long has that thing been emerging?!?) ANSI standard. For instance, it now has function prototypes.

I can't really hold it against a C compiler developer who doesn't support the entire ANSI standard yet, since there's always the chance that the standard could change *again* before it's finalized.

As with WATCOM C, I haven't had a

chance yet to benchmark Desmet C 3.1, but it's on my list.

Zortech C++

I've felt kind of bare talking about this package without having seen it. Well, it's finally here! The first true C++ compiler for PCs arrived a week or so before I left for SOG.

The manual is light-years ahead of the old Datalight Optimum-C manual (the compiler was written by the same fellow who wrote Datalight). Zortech has put some effort into the manual, and it shows. For instance, there are examples on how to use every library function.

They even include a C++ tutorial. While it'll give you a start, it won't teach you the fine points of object-oriented programming.

I recommend the book *An Introduction to Object-Oriented Programming and C++*, by Richard S. Weiner and Lewis J. Pinson (Addison-Wesley, 1988). I recommend it over Bjarne Stroustrup's *The C++ Programming Language*, which originally defined the language. It's good to read both, but the Weiner & Pinson book is more clearly written, and has better descriptions of the theories behind object-oriented programming.

C++ is much more than just "C with classes"; it is a wide-ranging enhancement of the standard C programming language. C is a poor language for multi-programmer projects involving multiple code modules. C++ rectifies this not only through classes and objects, but also by giving the module writer control over external access to his or her functions and data types. This is known as "abstraction," and it's a common component of other recently-developed languages like Modula-2.

The Zortech package includes everything. There's an editor, linker, librarian, object module disassembler, and make, along with a reasonably good sized library (with, of course, graphics functions). The only item missing is a debugger.

Although Zortech C++ is compatible with Microsoft's CodeView, you can only look at global variables and source code; local variables cannot be accessed by name. This isn't Zortech's fault — Microsoft hasn't released the formats of the special object records used for tracking local variables.

I've found several minor bugs. The installation program will only install on a hard disk if it is drive C. I solved this problem on my system by using the MS-DOS SUBST command, as in:

```
SUBST C: E:\
```

Some header files don't get copied, and one of the debugging libraries ends up in the wrong directory. There were also a few typos and formatting glitches in the manual.

Zortech is already working on the problems. According to Joe McIsaacs at Zortech, rather than making users wait for the next release, they are printing up corrected manuals and new diskettes and sending them free-of-charge to all registered owners.

C++ is much more than just "C with classes"; it is a wide-ranging enhancement of the standard C programming language.

Now if everyone else would be that responsive ...

Object-Oriented Microsoft C?

Microsoft was represented at SOG by Greg Lobdell, Product Manager of Microsoft's Languages Group. He confirmed the rumor that Microsoft is looking into adding object-oriented facilities to C and BASIC (object-oriented BASIC?). They're currently examining both C++ and Objective-C.

Hopefully, they'll pick C++; it would help define a standard. I would hate to see C end up a muddle mess, with each vendor coming out with a different set of object-oriented extensions.

MMC AD Programmer's Toolbox

This isn't a group of C-language functions to be linked with your programs; it's a complete set of utility programs which can make the life of a C programmer easier.

There are actually two toolboxes in the package: volumes I and II. Volume I contains 12 utilities; volume II has 11 more. There are several general utilities, like CAT (which copies files from standard input to standard output). However, it's the advanced utilities that really make the package worthwhile.

CLint (Volume II) does a syntactic check of a C source file. Lint utilities come from the UNIX world and provide better syntax checking than compilers normally do. For example, the following code fragment is in error:

```
int i1, i2;
printf("%d %s %c",i1,i2);
```

Most C compilers will not notice that there are two bugs in the above code. First, the third parameter should be a string; second, there should be a fourth (char). CLint catches these errors.

CPrint (Volume I) is a C "beautifier," meaning that it takes an existing C source file and rewrites it in a regular format. This can be very handy in an environment where several programmers are writing code using different ways of lining up brackets and statements. CPrint can convert all of the files to one format, making maintenance easier.

CXref (Volume II) generates a comprehensive cross-reference report of the contents of a C source file. It's much more sophisticated than the cross-reference generators that come with compilers. CXref will even report mathematical operators and C keywords. If you're working with a large source file, a utility like this is very useful.

PMon (Volume I) is a TSR which monitors a program's execution. There are other profilers on the market (both in the public domain and commercially.) PMon produces a list of system calls and a timing table. The timing table tells the user how much time was spent executing each function or line of source. This is an invaluable tool for optimization.

You can run PMon to see which functions or statements are using the most time. Then you can optimize them, or rewrite them in assembler, to improve your programs' performance. The CritPath utility in Volume II can provide further reports by analyzing the PMon data.

I really like this package. The documentation is well-written and the programs work as described. Each toolbox sells for $79.95, or you can get both for $130.

Choosing A C Compiler

During my talk at SOG, a number of people asked about choosing benchmarks. This seems to be a hot topic! Everyone seems to want to know which compiler is *best*. Out on the electronic mail circuit, at SOG, and in user's groups, programmers who want to work with C want to know what they should buy.

Choosing a compiler isn't easy; by the time of the next C compiler round-up (scheduled for the March/April 1989 issue), there will be at least 15 C compilers on the market. At least a half-dozen of these will make it into the "good to excellent" category, with the rest scrambling to catch up.

Whenever I get asked the fateful question, "Which compiler should I buy?," I offer guidelines rather than a specific recommendation.

First, Microsoft and Borland aren't the only compiler writers in the world. True, they put out good products; but there are also very good compilers from WATCOM, Zortech, Mark William's, MIX, and Manx. *(Editor's note: You might also be aware of JPI. I understand its programmers departed from Borland with the C package they were writing when Borland decided to purchase Wizard C.)*

What's your budget? Are you just going to dabble in C programming, or do you intend to produce professional programs? Do your applications use floating point calculations? Do you need fast programs or small ones?

In the C roundup, I made some general suggestions. Below I'll list the eight compilers I think are worth considering, along with my reasoning.

Zortech C++: For under a hundred dollars, you're not only getting one of the best optimizing C compilers around, but you can also move on to object-oriented C++ when you're ready. Zortech's support is very good and the compiler produces small, fast programs.

Microsoft C: If you've got the bucks (around 270 of them from a mail-order house), this is a good all-around choice. The big C compiler produces fast programs (albeit a bit large), and has every tool imaginable. The package also includes QuickC. The documentation is very good.

Microsoft QuickC: If you need Microsoft compatibility, or are just learning C, this is a good choice because of its integrated environment and debugger. Microsoft will be enhancing QuickC in the near future ...

Borland Turbo C: It's inexpensive (under a hundred dollars), compiles quickly, and has a nice integrated compiler/editor. There will be a debugger coming *real soon now*.

Mark William's Let's C: If you're doing UNIX programming, and want the best in UNIX compatibility, this is your choice. And, it's under $80 ...

Manx Aztec C86: For ROM-based applications, Manx gets my vote. Also, Manx supports C compilers for everything from CP/M machines to the Macintosh, making portability easier if you're working with several different micros.

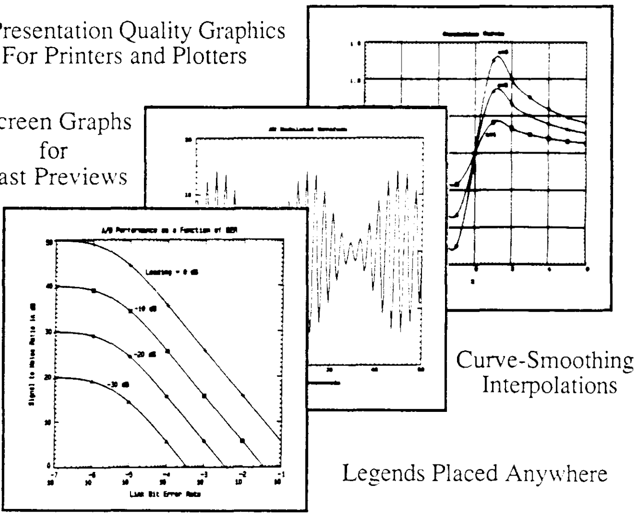
MIX Power C: What can you say against a working C compiler for 20 bucks? It's a bit unstable to use for professional development, but it's great for the hobbyist. They've ironed out most of the bugs, and if nothing else, you get a good C programming manual with a reasonable compiler thrown in.

WATCOM C: This is a powerful package on par with Microsoft's C. It optimizes well and has a slew of

SCIENTIFIC GRAPHICS

Presentation Quality Graphics For Printers and Plotters

Screen Graphs for Fast Previews



Curve-Smoothing Interpolations

Legends Placed Anywhere

Built-In Editor   Auto/Manual Scaling   Log/Lin/SemiLog

An Indespensible Tool For Technical Professionals

GraphStar

System Requirements: IBM-PC, XT, AT or Compatible running DOS 2.0 or higher. Printer Graphs require Epson EX, FX, JX, RX, HS; Star Gemini, Radix, SD, SG, SR; IBM Graphics; or compatibility with one of the above. Plotter graphs require HP-GL compatibility.

Scientific Software Solutions

P.O. Box 956, Dept. M, Valley Forge, PA 19482
For Technical Information: (215) 269-0198

Reader Service Number 60

CALL FOR FREE CATALOG

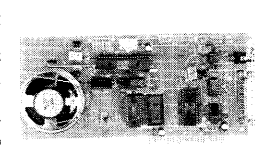
ALL SALES SUBJECT TO THE TERMS OF OUR 90 DAY LIMITED WARRANTY. FREE COPY UPON REQUEST.

TEXT TO SPEECH BOARD!

PC/XT COMPATIBLE. MAKE YOUR COMPUTER TALK!

A VERY POWERFUL AND AMAZING SPEECH CARD. USES THE NEW GENERAL INSTRUMENTS SPO256-AL2 SPEECH CHIP AND THE CTS256A-AL2 TEXT TO SPEECH CONVERTER.

THIS BOARD USES ONE SLOT ON THE MOTHERBOARD AND REQUIRES A COM SERIAL PORT. BOARD MAY ALSO BE USED IN A STAND ALONE ENVIRONMENT WITH ALMOST ANY COMPUTER THAT HAS A RS232 SERIAL PORT. FEATURES ON BOARD AUDIO AMP OR MAY BE USED WITH EXTERNAL AMPS. DEMONSTRATION SOFTWARE AND A LIBRARY BUILDING PROGRAM ARE INCLUDED ON A 5¼ INCH PC/XT DISKETTE. FULL DOCUMENTATION AND SCHEMATICS ARE ALSO INCLUDED.

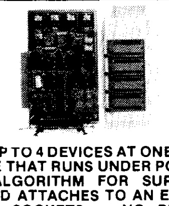


NEW! PRICE CUT! **$69.95** ASSEMBLED & TESTED

NEW! IC TESTER! $149.00

SIMILAR TO BELOW EPROM PROGRAMMER. PLUGS IN TO YOUR PC OR XT. TESTS ALMOST ALL 14, 16, AND 20 PIN 74XX SERIES. INCLUDES STANDARD POWER, "S" AND "LS" DEVICES. ALSO TESTS CD4000 SERIES CMOS. SOFTWARE INCLUDED CAN EVEN DETERMINE PART NUMBERS OF MOST UNMARKED AND HOUSE NUMBERED DEVICES WITH SIMPLE MOD. THIS UNIT CAN ALSO TEST 6.4K AND 256K DRAMS! WITH MANUAL AND SOFTWARE: $149. PERFECT FOR SCHOOLS.

PC/XT EPROM PROGRAMMER $169



ASK ABOUT OUR NEW PAL PROGRAMMER!

★ LATEST DESIGN ★ PROGRAMS UP TO 4 DEVICES AT ONE TIME ★ FEATURES EASY TO USE MENU DRIVEN SOFTWARE THAT RUNS UNDER PC OR MS-DOS. ★ USES AN INTELLIGENT PROGRAMMING ALGORITHM FOR SUPER FAST (8X) EPROM BURNING. ★ THIS PLUG-IN BOARD ATTACHES TO AN EXTERNAL MINI CHASSIS CONTAINING 4 TEXTOOL Z.I.F. SOCKETS. ★ NO PERSONALITY MODULES REQUIRED ★ AUTOMATIC VPP SELECTION: 12.5V, 21V, OR 25V. ★ EPROM DATA CAN ALSO BE LOADED FROM OR SAVED TO A DISKETTE. ★ PROGRAMMING SOFTWARE SUPPORTS: 2716, 2732, 2732A, 2764, 2764A, 27128, 27128A, 27256, 27256A, 27512, AND 27512A. ★ ASSEMBLED AND TESTED, BURNED. IN WITH MANUAL. $169 WITH SOFTWARE.

JUST RECEIVED. SAME AS ABOVE PROGRAMMER, BUT PROGRAMS 8 UNITS AT ONE TIME - $299.

Digital Research Computers

P.O. BOX 381450 • DUNCANVILLE, TX 75138 • (214) 225-2309

TERMS: Add $3.00 postage. We pay balance. Orders under $15 add 75¢ handling. No C.O.D. We accept Visa and MasterCard. Texas Res. add 6-1/4% Tax. Foreign orders (except Canada) add 20% P & H. Orders over $50 add 85¢ for insurance.

Reader Service Number 32

utilities. The library is good, and the floating-point performance is amazing. If you're doing a math-intensive project on a PC without a math coprocessor, WATCOM is an especially good choice.

Book Of The Month

The Society of Mind, by Marvin Minsky ($9.95, Simon & Schuster, 1986). Marvin Minsky is one of the founders of the Artificial Intelligence Laboratory at the Massachusetts Institute of Technology, and one of the best minds to ever think about thinking.

This book consists of a series of one-page essays, each building on the previous ones, which attempt to lead the reader into a better understanding of how we think. And *why* we think. It's the most fascinating book I've read in years. Anyone who is interested in AI, or who just wants to understand people a bit better, should read this book.

Drop Me A Line ...

... either on the Micro C RBBS, or at the P.O. Box listed at the beginning of the article, or by phone. If you use FidoNET, you can send NetMail to "Scott Ladd" on node 1:104/47.

I want to hear what interests you. Should there be more coding examples? More or less industry news? Is there a product you'd like to hear about?

I'll C you later!

Major Products Discussed

WATCOM C v6.5
**WATCOM Products Inc.
415 Phillip Street
Waterloo, Ontario
Canada, N2L 3X2
(519) 886-3700**

C Ware Desmet C88 v3.1
**C Ware Corporation
P.O. Box 428
Paso Robles, CA 93447
(805) 239-4620**

Zortech C++ v1.0
**Zortech Inc.
366 Massachusetts Avenue
Arlington, MA 02174
(617) 646-6703**

Programmer's Toolkit, Volumes I and II
**MMC AD Systems
Box 360845
Milpitas, CA 95035
(408) 263-0781**

**Microsoft Corp.
16011 Northeast 36th Way
P.O. Box 9097017
Redmond, WA 98073-9717
(206) 882-8080
(800) 426-9400**

Aztec C
**Manx Software Systems
One Industrial Way
Eatontown, NJ 07724
(800) 221-0440**

**Jensen & Partners International (JPI)
1101 San Antonio Rd., Suite 301
Mountain View, CA 94043
(415) 967-3200**

♦ ♦ ♦

ICs PROMPT DELIVERY!!!
SAME DAY SHIPPING (USUALLY)
QUANTITY ONE PRICES SHOWN for AUG. 23, 1988

OUTSIDE OKLAHOMA: NO SALES TAX

640K MOTHERBD UPGRADE: Zenith 150, IBM PC/XT, Compaq Portable & Plus; hp Vectra

DYNAMIC RAM			
SIMM	1048Kx9	85 ns	$550.00
1Mbit	1048Kx1	100 ns	38.50
41256	256Kx1	60 ns	14.50
41256	* 256Kx1	80 ns	13.75
41256	256Kx1	100 ns	13.75
51258	* 256Kx1	100 ns	12.95
41256	256Kx1	120 ns	12.75
41256	256Kx1	150 ns	11.75
41264	+ 64Kx4	120 ns	18.95
EPROM			
27C1000	128Kx8	200 ns	$37.50
27C512	64Kx8	200 ns	13.95
27256	32Kx8	250 ns	7.25
27128	16Kx8	250 ns	6.60
STATIC RAM			
43256L-10	32Kx8	100 ns	$18.95
6264P-12	8Kx8	120 ns	13.50

*STATIC COLUMN DRAM
+2-PORT VIDEO RAM
80387-16 80387-20
80287-8 $245.00
8087-2 $160.00

OPEN 6½ DAYS, 7:30 AM-10 PM: SHIP VIA FED-EX ON SAT.

WE EXPORT ONLY TO CANADA, GUAM, PUERTO RICO & VIRGIN ISLANDS

SAT DELIVERY INCLUDED ON FED-EX ORDERS RECEIVED BY:
Th: Std Air $6/3 lb
Fr: P-1 $10.25/1 lb

MasterCard/VISA or UPS CASH COD
Factory New, Prime Parts μP∞
MICROPROCESSORS UNLIMITED, INC.
24,000 S. Peoria Ave.,
BEGGS, OK. 74421
(918) 267-4961
No minimum order. Please note that prices are subject to change. Shipping & insurance extra, & up to $1 for packing materials. **Orders received by 9 PM CST can usually be delivered the next morning, via Federal Express Standard Air @ $6.00, or guaranteed next day Priority One @ $10.25!**

Reader Service Number 37

Explore Neural Net Technology $87.95

Connections: The Traveling Salesman is derived from the classic Hopfield-Tank neural net model for solving the combinatorially explosive traveling salesman problem. A previously undiscovered model parameter is identified. A new speedup based on markers for city-position pairs is introduced. Recent research on biological systems contributes a new method for better convergence of the solution.

Connections is written in PL/D, an innovative systems language reviewed in July *Computer Language.* You get a copy of PL/D and PL/D user manual with *Connections* so you have all you need to experiment with the program. Self-compiling source of PL/D is also available for $75.00 additional.

DOS 2.0, or above; 256K PC, AT, compatible, or PS/2. 5.25 or 3.25 inch diskette. Add $5.00 shipping ($12.50 outside North America) and sales tax in California.

DAIR Computer Systems VISA MASTER CARD 3440 Kenneth Drive Palo Alto, CA 94303

(415) 494-7081

Reader Service Number 90

86 WORLD

Yet Another Travelog ...

(From Out Of The East ... Somewhere)

By Laine Stump
℅ Redhouse Press
Merkez PK 142
34432 Sirkeci
Istanbul, Turkey

If you wondered what Laine goes through to write a column while camping, caving, busing, consulting, eating, and sleeping — (hey, it's a hard life), then read on. If you'd rather skip to the code for this issue, just skip to the end. The very end. (And if Laine calls and tells me the column is on the bus, I'll believe him. Next time.)

So you think it looks easy to write a magazine column, do you? You just sit down in front of the screen and the ideas flow swiftly out your fingertips and onto a disk which you exchange for large stacks of money. Piece of cake. Anybody could do it. Good way to earn a few extra bucks. Doesn't take much time at all ...

"That sounds like fun, Laine, but I'm sorry to say that I'm already planning to go caving in Kastamonu province next week. If you can't find enough people to go floating, you're welcome to come along with us."

I was on the phone with my friend Oral. Trying once again to get up the necessary five for a boat trip, I was running into snags. "If you like, we can take the boat along. There's one river that might be big enough, and it's an awesome canyon. We could take off from the cave camp for a day or two and try floating it. The camp's going to be quite large, so we'll have no trouble finding people."

Four days later I was trucking into the sunrise in a Nissan crew cab with Sami (Oral's brother-in-law and the owner of the Nissan) and Ceasar (Sami's mother's golden retriever). I was leaving a lot of work behind in the city but, hey, it's Kurban Bayram (a Muslim Holiday)! The whole country is on vacation. Why not me too?

While Ceasar wiped his wet nose against the back of my neck for the twentieth time, I quietly filed away a list of "Things To Do After Vacation." Let's see, I need to put Turkish on that Wyse monitor for Onur. I need to unsolder the character generator chip from my Zenith. I need to go to Ankara and show my Toshiba to that guy who's interested in buying it. I need to contact Infinite about Turkish fonts for PostScript printers. I need to write a column for *Micro C* ... Oh God! Not *another* one!

In Search Of A Topic

What can I write about? I've now been successful in publishing just about every piece of code I've ever written. At least the interesting pieces. And some of the boring pieces. And even some that I stole from somewhere else (SShhhh!). What am I going to write about????

With Ceasar nudging my back, competing for window space to flap his tongue in the breeze (much like his namesake, and like most politicians of today as well), I tried to bring up a mental picture of everything I hadn't already written about. Blank. I'll have to try again tomorrow.

We arrived in the base camp deep in the luscious green forests of the Western Black Sea Coast the next morning and, after setting up my tent and meeting a few of the other campers (members of the Bosphorus University Cave Explorers Society), I tried to think of things to write about. Still no luck. Maybe after lunch.

While we were four-wheeling down the high mountain roads, swimming in rock studded canyons (not enough water to boat), and (of course) photographing the many varieties of unique looking mushrooms in a newly discovered cave, I was constantly (well, sometimes) trying to think of topics to write about before my upcoming deadline.

Well, I hate to let this out, but it's pretty damn difficult to come up with article ideas for a computer magazine while frying bread over a gasoline backpacking stove in the middle of a beautiful green meadow over 50 miles from the nearest paved highway. Maybe I should see if there are any openings at *Outside Magazine*.

I asked some of the others for ideas.

"How about 'Caving as Recreational Therapy for Computer Scientists?'"

"I know. You can write about a computer controlled camp dinner menu creation system. Of course it would ignore all requests to put sujuk (a disgusting spiced sausage) on the menu."

"Show how to write a program to take all the measurements we make and generate an accurate map of the cave automatically."

"Wait a minute! Let's get back to this idea of sausage elimination."

No luck there. We had the sujuk, and we had to eat dinner. But I didn't have the necessary equipment to do the development work (my new machine doesn't run on batteries). Besides, you don't need a computer program to eliminate sujuk from the menu.

So I gave up for awhile. Hung my sleeping bag out to dry in the morning. Dug the Landrover out of a clay bank in the afternoon. Dashed across the mountain tops clinging perilously to the roll bar on the back of the Nissan. Sat around the campfire scrounging "taste tests" from the evening meal as it was being cooked. Refused to wash my hair until just two days before the end of camp.

Still In Search Of A Topic

Back in Istanbul I was in true form. "What's up, Laine?"

"Oh, I've got this stupid magazine article to write and I can't think of anything."

"If you sit down for a few hours something will come."

"Yeah, maybe. But I've got to go to Ankara and sell my old computer tomorrow. Maybe I'll think of something while I'm there ..."

Off on the night bus to Ankara. My seatmate was a student in dentistry at Hacitepe University in Ankara. We talked about how rich the dentists get in the U.S. and how Turkish dentists aren't compensated nearly enough for their efforts. No *MC* article ideas there, but at least he told me about a new Mexican restaurant just opened by a friend of his in Ankara (the first Mexican restaurant in Turkey!).

We parted in the morning with an exchange of phone numbers and a promise to get together and eat Mexican food ("It would be very useful for you to go there. None of us know what Mexican food is supposed to taste like. They just put something on the plate and if it's hot we figure it must be right. They really need an expert's opinion.") Still no ideas for an article. I hoped I'd come up with something while working in Ankara.

It was the normal scene in Ankara.

"Laine, did you ever find the IC I need for my power supply?"

"We need some help making those new 30 meg drives work right."

"We're converting these programs from Turbo Pascal 3.0 to 4.0 and we can't figure out what to do with this inline assembly code that does an indirect call to ConOutPtr."

"I just imported these printers from France for next to nothing, but they don't have a Turkish character ROM. Do you suppose ..."

In two days I managed to: put the Turkish character set on a Wyse high resolution monochrome card and a no name Japanese printer; teach a technician (who doesn't speak English) how to install and format Miniscribe 8438 drives with OMTI 3527 RLL SCSI bus controllers (fast, reliable, and cute); send a FAX to PC Tech asking what happened to the 20 5380 SCSI chips we asked for; inquire once again into the disposition of my reimbursement for bus and airplane tickets from last January; and even get a down payment for the sale of my Toshiba T1100. Still no article ideas.

Oh, I didn't quit thinking about it. I asked everyone I met.

Ergun said, "Why don't you forecast what's going to happen in the industry in the next few years?"

"Are you kidding? Only Jerry Pournelle and John Dvorak are godlike enough to do that."

"How about forecasting what you would *like* to happen in the next few years?"

"If I did that, everyone would think I was just complaining (again)."

"I guess you could always write about your caving trip. Judging from that silly thing you wrote last year, you'd probably get away with it."

Back to the main menu.

In Search Of A Topic, 3D

Another night bus back to Istanbul.

"I'm really going to do it this time. I'll come up with a humdinger, all time showstopper of an article. This one's going to knock their socks off. Yeah. That's it. Their socks. And their monogrammed slippers, too. They'll be wiping Jolt Cola off the floor with their bandanas after this article knocks them out of their seats. Yeah. Jolt Cola. That's the ticket. Jolt Cola ..."

The potholes on the Istanbul highway rocked me off into dreamland, as I feebly attempted to squeeze an article idea out of my last two months.

I awoke with a start, brought to my senses by the putrid smell of burning rubber, and realized that the bus was stopped. As the moustached man in the next seat snored on in oblivion, the steward turned on the PA system "Respected passengers: One of our tires has blown out." It was 2 a.m.

Off the bus to check out the situation. There was a crowd of about 15 men gathered around one of the rear wheels, trying desperately to survey the situation with only the aid of a small cigarette lighter. I climbed back into the bus and returned, much to the amusement of the crowd, with my Tekna Lite 2 waterproof flashlight.

"Is he a tourist?"

"No, I'm not a tourist."

"Hey, you know Turkish! Does that thing run on NiCads, or what?"

An hour and a half later, the tire replaced and the brake drum cooled with several bottles of water, I sat back in my seat. My seatmate was just waking.

"What happened? Did the brakes lock?"

"Yeah. Just after the tire blew out over an hour ago."

"Oh."

He was immediately back in his snoring stupor. Obviously he didn't have to come up with a column for a computer magazine.

Now where was I? Let's see ... Computer controlled Dorito production in developing nations ... With such an exciting topic to think about, sleep arrived in my frontal lobes (or wherever it is that sleep arrives) without much provocation.

I was awakened by the steward three hours later as the bus motored through the first glimmer of dawn, just pulling into Izmit at the east edge of the Sea of Marmara. "Would you like some tea?" he asked as he, not unpolitely, shoved the glass under my nose.

It was the aroma that awoke me. My most pleasant awakening in months. Not counting last Saturday morning waking up in the dewy 6 a.m. mist next to a dormant campfire, doing my best to not wake anyone else as I started the fire for morning tea. Yeah. Morning tea.

What if I wrote about the possibilities of tea packaging machines that automatically tested radiation levels of tea (a big question down here just south of Chernobyl) and automatically put the bad tea into a special hopper for delivery to the Nuclear Energy Commission. Or maybe the effects of high tea consumption on the writing energy of witless young computer journalists ...

Nah. Nothing has substance yet. If that OS/2 kit would just show up ... And the other Meg of RAM. But then one of the guys at the caving camp works for Lotus in Boston, and he told me that even 2 megs isn't enough to do anything useful with OS/2 ... So, like I said, if that other 3 megs of RAM would show up ...

Jeez, I can't even get a manual for Turbo Pascal 4.0! What am I doing talking about an OS/2 Developer's Kit. I should be trying for the easy stuff first. Sometimes it's not easy being a freewheeling computer "samaritan." Maybe I should try writing again about getting jobs overseas. That might prompt me to go do some work myself so I would have enough money to buy this stuff with cash instead of trying to beg it under the auspices of doing magazine reviews.

The bus arrived in Istanbul during the morning rush hour so it wasn't allowed to drive downtown and drop me off at the bus company's office near my house. Instead, we pulled off at an overpass to transfer to a smaller shuttle bus for our final trip into the city.

As the cargo hold was opened, I noticed that my computer was no longer sitting on top of its six inch foam pillow. "What have you done?!?" I asked the steward. "I said that my computer had to *always* be on the pillow! It's very delicate!"

"It's okay. I just moved it half an hour ago, at the last stop."

If damage was done, it was already done. I was too tired to explain to him that the little black bag I was so concerned about cost more than his wages for an entire year, and that it didn't matter if it was off the pillow for five hours or five minutes, it only took one good bump in the road to clobber the winchester drive. I tossed my things into the service bus and headed home for a shower.

Hmmm. Maybe I could write about the bad things that can happen when you're working overseas. The misconceptions. The disorganization. The equipment ruined by ignorance. The frustrating delays and feelings of falling off the edge of the earth as soon as you take off from JFK International.

The people who don't trust computers and insist on entering all the numbers in the spreadsheet in alpha mode and adding them up by hand. The executives who don't take you seriously because you wear T-shirts and blue jeans to the office in a country where even farmers sometimes wear ties and suit coats in the field. Because you carry a Caribou Mountaineering backpack instead of a briefcase, and because you are under 45. No, that would be too negative. I complain about negative people too much to be one. At least to publicly be one.

In Search Of A Topic, Part IV

At home I hit the button on my answering machine and got the message from Lois and Elaine, calling from Van in Southeastern Turkey. "We're still planning on meeting you in Trabzon at 4 p.m. on the 10th. Onok and his friend are coming, too. We decided that we should meet at the Tourism Information Office instead of the City Hall, though."

Great. I've got three days to unpack, pack, and get on the bus for an 18 hour ride out to Trabzon. The boat is at Sami's mother's house and it needs repairing. I need to seal the seams of my tent, wash all my clothes and duffles, take inventory, call Tarsus and have them send up the boat adhesive from Jack's refrigerator. And I still have to think of a topic for the *Micro C* column. The seagulls screaming from the top of the mosque across the street

Performance and versatility for your CP/M or MS-DOS computer

QP/M

QP/M by MICROCode Consulting
Fed up with the message **"BDOS error: R/O"**? With QP/M, you'll never lose another file because you changed a diskette. QP/M offers full CP/M 2.2 compatibility with outstanding performance and more commands WITHOUT eating up precious program space. Get such features as automatic disk relogging, simple drive/user selection using either a colon or semi-colon, 31 user areas, drive search path, multiple program command line, archive bit maintanence, and transparent time/date stamping; all in the same space as CP/M 2.2. Installs from a convenient customization menu, no software assembly required. Bootable disks available with CBIOS for Kaypro, Xerox (8" or 5¼", -1 or -2), & BBI.

QP/M Operating System, bootable - specify system	$ 64.95
QP/M without CBIOS (installs on any Z80 system)	$ 49.95

Networks

QP/M Network File System by MICROCode Consulting
QP/M Network File System is an efficient local area network allowing up to seven CP/M computers to share peripherals and data resources.

- Transparent operation at speeds up to 11,000 bytes/second in syncronous mode
- Speeds of up to 1,920 bytes/second in asyncronous mode
- Local/remote disk drive and printer support
- Remote peripheral support for modems and real-time clocks
- All stations need not be on the network even though connected
- Local drive access protection and control
- Simple menu oriented configuration utility
- Extended DOS calls are provided for addition of custom network utilities.

Works with interrupt driven Z80 systems such as Xerox 820, Kaypro (KayPLUS & Advent ROMs), Eagle, and other computers running QP/M, or CP/M 2.2

QP/M Network File System	$ 39.95

Hard Disks

Need more speed and storage on your system?
Improve the productivity of your Z80 computer with a hard disk.

HDS Host Board
This daughter board provides a convenient interface for connecting a Western Digital WD1002-05 hard disk controller to your computer.

- Plugs into the Z80 socket, no other wiring required
- 40 pin interface for a WD1002-05 (or HDO) controller board
- Switch selectable I/O port addressing
- Comes as bare board or assembled & tested
- Kaypro '84 host board also available

Winchester Connection by MICROCode Consulting
The most simple and comprehensive hard disk software package available for CP/M.

- Designed for use with the WD1002-05 controller board
- Works with one or two hard disks - 5 to 64 meg
- Menu installed, no software to assemble
- Complete hardware tests and error handling
- Automatic swap, for warm boots from hard drive
- Software drivers install above or below CP/M
- Allows custom partition sizes and mixed drive types
- Independent block and directory sizes on each partition
- Includes manual, format, test, park, and swap utilities

Winchester Connection Software only	$ 39.95
HDS Board with Winchester Connection Software	$ 79.95
HDS Bare Board with software	$ 59.95
HDS Board, WD1002-05, and software	$245.00

Call or write for other pricing options

WD1002-05 HARD DISK CONTROLLER BOARD by Western Digital

- Standard ST506 drive interface
- Same size as standard 5¼" drive
- 40 pin interface to host computer
- WD2797 floppy disk controller interface on board
- Can control up to three hard drives
- Direct replacement for **Kaypro 10** controller

WD1002-05 Controller Board	$185.00

Other Western Digital boards available

Prices subject to change without notice. VISA and Mastercard accepted. Include $5.00 shipping and handling, $7.50 for COD, UPS-Blue or RED Label additional according to weight. Please include your phone number with all correspondence.

Kaypro

KayPLUS ROM Set by MICROCode Consulting
Want more performance and flexibility from your Kaypro? With the KayPLUS ROM set you can have the advantages of a Kaypro 4 or 10, even on your Kaypro 2.

- Install up to four floppies and two hard drives
- Boots from floppy or hard disk
- Supports 96 TPI and 3½" disk drives
- Can use any ST506 type hard drive - 5 to 64 Meg
- 32 character type-ahead keyboard buffer
- Automatic screen blanking (not avail. on 83 series)
- 12 disk formats built-in, unlimited configurable
- Full automatic disk relogging with QP/M
- Internal real-time clock support
- No software assembly required

Includes manual, format, configuration, diagnostics, sysgen, diskette customization utility, AND hard disk utilities. Available for '83 and '84 series Kaypros.

KayPLUS ROM Set, specify model	$ 69.95
KayPLUS ROM Set with QP/M	$125.00

Parts and accessories for the Kaypro

Kaypro 2X Real-time Clock parts kit	$ 29.00
Kaypro 2X Hard disk interface parts kit	$ 16.00
Kaypro 10 or '84 series Hard Disk host board	$ 49.00
Kaypro four drive floppy decoder board	$ 35.00

Complete parts and repair services available

Xerox 820

PLUS2 ROM and X120 Double Density Board by MICROCode Consulting and Emerald Microware
About had it with single density diskettes on your Xerox 820-1? Get unsurpassed versatility with our X120 Board and PLUS2 ROM package.

- Run up to four floppy disk drives at once
- Mix 8" and 5¼" at the same time
- Software compatible with Kaypro and Xerox 820
- Built in drivers for most serial and parallel printers
- Get mini-monitor functions and auto-boot capability
- 19 built in disk formats, including Xerox and Kaypro
- Includes custom disk format definition program
- Banked ROM BIOS for more space in your TPA
- Composite video adaptor on X120 board
- Runs 48 TPI diskettes on 96 TPI drives
- Supports real time clock from Z80-CTC
- Works on the Xerox 820-1 and Big Board I
- Both ROM and X120 board are required for operation

PLUS2 ROM Set and X120 Board A&T	$114.95
PLUS2 ROM Set and X120 Bare Board	$ 49.95
PLUS2 ROM Set only	$ 39.95
120 Bare Board only	$ 15.00

★★★ **Special** ★★★ 2 boards for $25, 5 for $50
Other kits, parts, and packages available

Parts and accessories for the Xerox 820

Xerox 820-2 CPU Board - new	$ 75.00
Xerox 820-2 Floppy Controller board - new	$ 65.00
Xerox 820-2 CPU board w/ Floppy Controller	$125.00
Xerox 820-1 CPU board - new	$ 75.00
Xerox 820 complete high profile keyboard	$ 65.00
Xerox 820 bare high profile keyboard - new	$ 25.00
Xerox 820 5¼" drive cable	$ 9.00
Xerox internal video cable w/brightness control	$ 9.00
Xerox 820 power supply	$ 35.00
Power connector, specify board or cable	$ 2.50
Xerox parallel printer interface cable	$ 35.00
Dual Half Height 5¼" Disk Drives - DSDD, in cabinet with standard Xerox cable	$265.00

Complete parts and repair services available

EMERALD MICROWARE

P.O. Box 1726, Beaverton, OR 97075

VISA **(503) 641-0347** MasterCard

30 day money back guarantee on all products.

IBM PC

CP/M, NorthStar, Macintosh, Apple II, MS-DOS, and PS/2 - Don't let incompatible diskette formats get you down, read them all with your PC.

UniForm-PC by MicroSolutions
How often have you wished you could use your CP/M diskettes on your PC? Now you can access your CP/M disks and programs on your MS-DOS computer just as you would a standard MS-DOS diskette. Once the UniForm driver is installed, you can use standard DOS commands and programs right on your original diskette without modifying or copying your files. UniForm-PC allows you to read, write, format, and copy diskettes from over 275 CP/M and MS-DOS computers on your PC, XT, or AT. With UniForm-PC and the Compaticard, you can use 5¼" high density, 96TPI, dual format 3½" (720k/1.44 meg.- PS/2), and even 8" drives.

UniForm-PC by MicroSolutions	$ 64.95
Uniform for Kaypro and other machines	$ 64.95

CompatiCard by MicroSolutions
Meet the CompatiCard, THE universal disk drive controller card. This half card will let you run up to 16 disk drives (4 per CompatiCard) on your PC or XT, including standard 360K, 96 TPI, high density (1.2 meg, dual speed), 8" single or double sided (SD or DD), and dual format 3½" drives (720k/1.44 - PS/2). The combinations are almost unlimited. Comes with its own MS-DOS driver and format program for high density and 3½" diskettes. Use it with UniForm-PC for maximum versitility. 8" adaptor and additional cabling available.

CompatiCard Board	$169.95
CompatiCard with UniFORM-PC. . ★★ **Special** ★★. .	$225.00
CompatiCard with UniFORM-PC & high density or 3½" drive . . . . . ★★★ **Special** ★★★	$350.00

MatchPoint-PC by MicroSolutions
The MatchPoint-PC board for the PC/XT/AT works with your standard controller card to let you read and write to NorthStar hard sector and Apple II diskettes on your PC. INCLUDES a copy of the UniForm-PC program, as well as utilities to format disks, copy, delete, and view files on Apple DOS, PRODOS, and Apple CP/M diskettes.

MatchPoint-PC Board	$169.95

MatchMaker by MicroSolutions
Now you can copy your Macintosh diskettes right on your PC/XT/AT with the MatchMaker. Just plug your external Macintosh drive into the MatchMaker board and experience EASY access to your 3½" Mac diskettes. Includes programs to read, write, initialize, and delete files on your single or double sided Mac diskettes.

MatchMaker Board	$139.95
MatchMaker w/External Mac Drive	$325.00

Frustrated because your PC can't speak CP/M?

UniDOS by Micro Solutions
Run CP/M programs on your PC? Of course. UniDOS is a memory resident program that can use the NEC V20 CPU chip to actually RUN your favorite 8080 programs. Use UniDOS with UniForm-PC, and automatically switch to CP/M mode as you log on your CP/M diskette. Switch to emulation mode to run Z80 code programs or for systems without a V20. UniDOS directly converts video and keyboard emulation for Kaypro, Xerox 820, Morrow, Osborne, VT100, and eight other displays. All standard CP/M system calls are supported. Note: The NEC V20 CPU is a fast, low power, CMOS replacement for the 8088 CPU chip that includes a full 8080 instruction set as well as the standard 8088 set. Systems using an 8086 may substitute a V30 chip.

UniDOS by MicroSolutions	$ 64.95
UniDOS w/UniForm and V20-8 chip	$135.00

UniDOS Z80 Coprocessor Board by MicroSolutions
This **8 Mhz. Z80H** half-card will run your Z80 and 8080 code programs at LIGHTNING speed on your PC or AT. Functions just like the UniDOS program, except NO V20 or emulation mode is required to run your programs. Now includes UniForm-PC!

UniDOS Z80 Coprocessor Card	$169.95

Reader Service Number 10

called into my mind an image of a dead albatross tied around my neck.

It was 8:20 a.m. Just over six hours since the tire blew up on Bolu pass. I grabbed a towel and headed for the shower but was stopped by the telephone ringing. "Hi, Laine. This is Ayfer. What are you doing this afternoon?"

"Oh, just resting," I lied.

After my shower I went to Redhouse Press to discover everyone else out on vacation. Fatih was there, though, and I asked him if he had any ideas of what I could write about.

"How about the things we've been doing to get the transfer between our machines and the typesetter working properly?"

"I don't think so. The readers would never understand why it took so much time to do something seemingly so easy. They don't care about the trials and tribulations of having an alphabet with extra characters not in the ASCII or ANSI standards. It just doesn't apply to them."

Arriving home at 9:30 p.m., I saw the huge pile of clothes on the bedroom floor and decided to give up for the moment. Maybe if I rested for half an hour I'd think of something. The bed seemed abnormally comfortable (not surprising, since I'd spent two of the last three nights on the bus).

My next moment of awareness was at 2:45 a.m. I stumbled to the bathroom and pried my contacts out of my eyes, cursing at my stupidity in once again falling asleep without removing them. I resolved to come up with a real mind bending topic first thing in the morning and crawled back to bed.

The next morning I borrowed Murad's car and drove up to Sami's mom's house to pick up the boat. Then out to the bus station to get the boat glue newly arrived on the overnight bus from Tarsus. Things were starting to pull together. "I should be able to leave tomorrow," I said, without much confidence.

Oh yeah. I forgot. I've got to do my taxes. I got a three month extension since I've been working overseas, but my three months will come due while I'm somewhere in the middle of a rapid on the Coruh River in Northeastern Turkey. I added that to my list and realized that maybe, for the first time ever, I would have to call Cary and tell her that I just couldn't write a column.

I waited until 2:30 a.m. (that's 4:30 p.m. in Oregon) and called Micro C.

"Hey Cary, I just can't make it. If you don't see anything by Wednesday, that means I couldn't come up with anything."

"Oh, that's fine. We'll just put another article in place of yours."

She almost sounded pleased!

Then I talked to Larry.

"Yeah, I tried to get out of writing a column once too, but Dave convinced me otherwise. You're lucky to be so far away. It's a bit tougher to be convinced from such a long distance."

Yea! My conscience is free! I don't have to write a column now. I can go down and get a bus ticket for Saturday. Just leave tomorrow. Just like that.

"Sorry, but all our buses to Trabzon are full until Tuesday." Okay, I can live with that. Maybe I'll write that column after all. But it has to have substance. I can take a look through my source disks and see if there's anything worthy. Let's see ...

Return Of In Search Of A Topic

A program to set the serial port to 38400 baud? No, I got that out of *BYTE*. A rewritten boot sector that handles any format of boot disk? That would have been useful back when I wrote it, in the DOS 2.11 days, but it's pretty worthless now, what with DOS 3.3 and all that. Besides, it's long.

How about a program to lengthen the timeout on INT 17h printer status to avoid "printer not ready" errors on slow printers? Nah. It doesn't really show anything new. Okay, then how about the low level procedures for that disk copy program I was going to write for Tony and never finished? Well, maybe ... What if I also threw in the SWAPDISK program that can make the B drive appear to be A or vice versa? Well, maybe ... I guess it's better than nothing.

Alright then. I'm ready. Where's my coffee cup? I need some drugs in my system if I'm going to do this. Let's get the CD player hooked up and some good Indian tabla music going; do this up right ...

Disks

Okay, so the topic is: Low level disk functions on PC compatibles. Has this been covered before? Probably. As usual, I won't bother telling you about the details you can learn from *The Pink Shirt Book*. I'll just give you some code fragments that you can use in your own programs.

Reading And Writing Tracks

Awhile back I was going to write a disk copy program that didn't bother rereading the disk after making the first copy. But like usual, I got so bogged down thinking of the ultimate user interface that I never finished it. I did write the procedures to read, format, and write the disks, though. (See Figure 1.)

The functions needed to access the disk at this low level are fairly easy to use, thanks to the biosdisk() function in bios.h of the Turbo C library. readtrack() and writetrack() just put the biosdisk() function into a retry loop. This is necessary because many BIOSes don't wait for the disk to become ready, so they often return a "Not Ready" error on the first call. I've found that three retries usually ensure reliable error reporting.

formattrack() is a bit more complicated, but not much. First, it must build a table of sector header information with each entry being four bytes: track, head, sector, sectorsize. Second, a verify operation is performed on each track after formatting before giving it a passing mark. Note that all the constants for SECSIZE, NUMSECS, etc., are for standard 360K diskettes. If you want the program to be general, you should make these constants into parameters which are passed as arguments to the functions. Hey, these are just examples. What do you expect?

Since an entire disk would be much larger than the maximum limit of 64K for a single data item (in anything but HUGE memory model), I decided to represent a disk as an array of NUMTRACKS pointers to one track worth of information. This data structure was defined in three steps to make it easier to understand.

First I made a typedef called TRACKBUFFER, which is an array large enough to hold one track worth of data. Then I made the typedef BUFPTR, which is a pointer to one track of data. Finally I declared the variable "data," which is an array of BUFPTR (i.e., an array of pointers to a full track of data). It would be possible to declare the same data structure with a single variable definition, but then probably nobody short of Kernighan or Ritchie would understand it. This way it's pretty obvious.

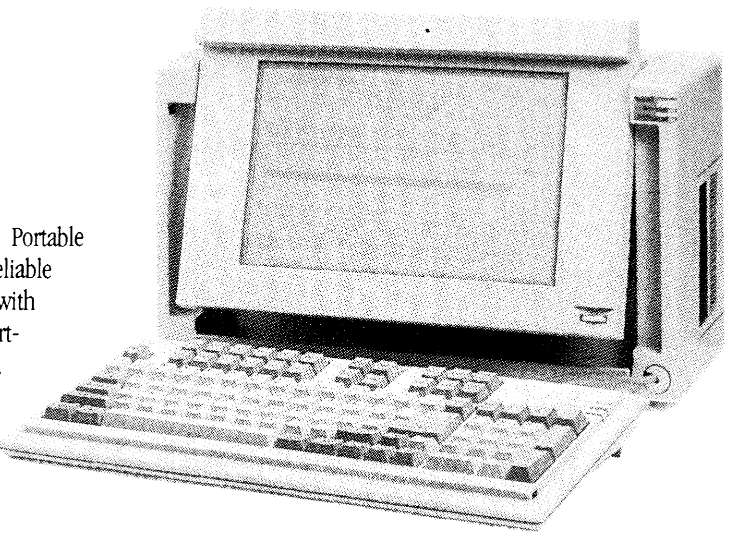
Reading And Writing Disks

Most of you can now easily see what you need to copy an entire disk. I see no reason for publishing Yet Another for()

McTek LCD Portable

(Including Hard Disk Only 19 lbs.)

RABBIT 286



The McTek Rabbit-286 LCD Portable combines the fastest, most reliable AT motherboard available with most visible full-size LCD portable screen on the market. Running at a switchable 8 or 10 MHz ∅ wait state, it includes a 20MB hard disk, 720KB 3½″ floppy drive, parallel & serial ports, Award 3.03 bios, 640k & turbo indicator LCD. The screen is a fantastically readable, electroluminescently backlit, 80-column by 25-line, high resolution 640x400 super twisted LCD with adjustable intensity and screen-angle. The screen size is 9.5″x6″. It's as readable as a CRT. You can also plug in a digital or analog color monitor or a digital or composite monochrome monitor. Included also is an external 5¼″ floppy port for reading and converting to 3½″ disks (5¼″ external drive w/case: $179 when purchased with LCD Portable). The McTek Rabbit-286 LCD Portable comes fully assembled with our one-year parts & labor guarantee, and sells for an amazing, complete price of only ***$1799!***

386-20 MHz W/1MB ***$2899!***

3 MB On-Board AT!

Our McTek 286A is the most integrated AT-compatible to date. It utilizes the highly regarded Chips & Technology chip set, and includes memory upgradable *on board* to 3 megabytes. No more worries about speed compatibility with expanded memory cards! The 8/10 MHz, ∅-wait state McTek 286A runs at 11.5 Norton SI, and an effective 13.2MHz on the Landmark test. Serial, parallel & game ports are all standard on board. With Award 3.01 bios, 640k, 200W power supply, Samsung amber monitor with Hercules-compatible controller, locking case, AT-style keyboard, 1.2MB drive, 20MB Seagate. Assembled & fully tested, with a full one-year warranty. Get in on the most advanced AT-compatible on the market, at the lowest price ever offered! *$1399!!*

XT Turbos & Supers

640k 4.77/8MHz and 4.77/10 switchable XT turboboards; two 360k floppy-disk drives with controller; one parallel, one serial and one game port; AT-style keyboard; clock, FCC-approved slide-case; eight slots; Hercules-compatible graphics card; amber monitor w/base; fully assembled and tested; one-year parts *and* labor warranty.

$729 **XT Turbo 4.77/8MHz Complete**

Superturbo 4.77/10 MHz Complete **$779**

McTek Systems, Inc. • 1411 San Pablo Avenue • Berkeley, CA 94702 • 415-525-5129

McTek

DISK DRIVES

Fujitsu 360k	$69
Fujitsu 1.2MB	$89
Teac	$75
Teac 1.2MB	$95
Toshiba 3½″ 720K	$99
Teac 3½″ 1.4MB	$129
Floppy controller	$22
20MB Hard Disk Kit	$279
30MB Hard Disk Kit	$309
ST-225	$215
ST-238	$239
ST-138 30MB	$399
ST-251 40MB	$369
ST-125 20MB 3½″	$279

PRINTERS

Citizen CD 120	$159
Citizen CD 180	$189
HPLASAR Serial2	$1699
Epson LX-800	$219
Toshiba 321 XL	$559
NEC P2000	$369

Call for prices of other brands

MODEMS

Everex int. 300/1200	$79
Everex 2400 external	$195
Everex 2400 internal	$179

MONITORS

Samsung amber	$79
Samsung EGA color	$359
Samsung RGB color	$259
NEC Multisync	$559
Sony Multiscan	$619
HGC-compat.mono card	$49
Color graphic card	$49
EGA Paradise 480	$149
VGA Paradise	$279
Genoa Super VGA	$299

MOUSE

Logimouse C7	$69

PC/XT

640k TurboMothrbrd	$85
10MHz TurboMothrbrd	$89
Multi I/O w/disk contrlr	$59
640k RAM card	$39
2MB Expansion card	$89
RS232 2-port card	$35
4-serial port card	$79
Game I/O card	$15
384k Multifunction card	$69
FCC-app. slide XT case	$29
150W power supply	$49
XT keyboard	$42
Clock Card	$19

PC/AT

McTek286-20MHz	$559
Baby McTek 286B-AT 8/10 O-wait	$269
McTek 286A O-wait 3MB 4 ports on board	$379
Multi I/O card	$55
Locking slide case	$59
200W power supply	$65
Enhanced keyboard	$59
WD HD/floppy controller	$129
3MB EMS (∅K)	$99

DESKTOP

MISC.

Kingtech CRT Portable Kits: XT/AT (power supply, case keyboard, monitor)	$380/$410
Eprom burner 4-socket	$139
LCD Portable	$799
AC power center	$25
AC power strips	$15
Diskette file box	$9
Printer or serial cable	$8
Archive Tape Backup 40MB	$299
386-20 MHz Desk Top	Call
286-20 MHz Desk Top	Call

Reader Service Number 42

Loop in the magazine. I'm sending in the readdisk and writedisk functions to *Micro C* though, and they will put them on the bulletin board or something. Have fun with them.

SWAPDISK

And what about that SWAPDISK program I talked about? Well, it's pretty simple, too. No sense in wasting paper on a listing. Just look for it on the bulletin board or on the Issue #44 disk.

I should tell you why I thought it necessary to write SWAPDISK, though.

Redhouse Press is using a program (PTS) sold by Compugraphic which can transfer word processing files to a diskette readable by a Compugraphic typesetter. PTS is pretty cheesy, but it works.

One of its problems is that the "Compugraphic Diskette" (i.e., the one that is in "Compugraphic format") can only be drive A or drive B. All of the Redhouse machines have 3.5 inch floppies in A and B. Compugraphic machines use 5 inch disk drives, so I wanted to put a 5 inch in as C. PTS, in the spirit of true "User Hostile" software, will not allow that.

I first tried using the ASSIGN program that comes with MS-DOS. Unfortunately, ASSIGN works at the DOS level, while Compugraphic's PTS program accesses the disk with ROM BIOS INT 13h calls (the same as our readtrack and writetrack procedures). What I needed was something to reroute disk calls at the ROM BIOS level.

It was really pretty easy. I just wrote a TSR (in assembly language) which captures all INT 13h calls and changes the drive number according to a routing table. This has been working nearly daily for over six months now.

I should say that, although in this case SWAPDISK is better than ASSIGN, there are still many uses for ASSIGN. ASSIGN works great for programs that access the disk only at the DOS level. It is especially nice since it lets you send requests to a winchester drive, which SWAPDISK cannot do. (SWAPDISK only works with floppy disks.)

What A Bunch Of Bull!

Can you believe this? I go through all that mental torture for over two weeks just to end up with a couple of silly little programming examples. The only thing that would be even more incredible is if they actually print it ...

Editor's note: Sorry Laine. If Larry can't get away with travelogs, neither can you. Be forewarned that at this very moment I'm at the Bend Trailways station waiting for the next bus to Istanbul. See ya.

♦ ♦ ♦

Figure 1 - Example C code for reading and writing tracks

```
#include <bios.h>

#define DISKREAD   2
#define DISKWRITE  3
#define DISKVERIFY 4
#define DISKFORMAT 5
#define NUMSECS    9
#define NUMBYTES 512
#define SECSIZE    2                /* 0=128, 1=256, 2=512, 3=1024 */
#define NUMTRACKS 40
#define NUMHEADS   2

typedef char TRACKBUFFER[NUMSECS*NUMBYTES];
typedef TRACKBUFFER *BUFPTR;
BUFPTR data[NUMTRACKS][NUMHEADS]; /*array of pointers to TRACKBUFFER*/

int readtrack(int drive, int track, int head, BUFPTR data)
    {
    int ct, success = 0;

    for (ct = 0; (ct++ < 3) && (!success); )
      success=(biosdisk(DISKREAD,drive,head,track,1,NUMSECS,data)==0);
    return (success);
    }

int writetrack(int drive, int track, int head, BUFPTR data)
    {
    int ct, success = 0;

    for (ct = 0; (ct++ < 3) && (!success); )
      success=(biosdisk(DISKWRITE,drive,head,track,1,NUMSECS,data)==0);
    return (success);
    }

int formattrack(int drive, int track, int head)
    {
    int *tptr;       /* tptr MUST HAVE sizeof(int) = 2*sizeof(char) */
    int   ct;  /* as well as low order byte being at lower address */
    int success = 0;
    static char formatinfo[NUMSECS][4] = { {0,0,1,SECSIZE},
                          {0,0,2,SECSIZE}, {0,0,3,SECSIZE},
                          {0,0,4,SECSIZE}, {0,0,5,SECSIZE},
                          {0,0,6,SECSIZE}, {0,0,7,SECSIZE},
                          {0,0,8,SECSIZE}, {0,0,9,SECSIZE} };

    tptr = (int *) formatinfo;
    for (ct = 0; ct++ < NUMSECS; )
        {
        *tptr = track | (head << 8);
        tptr += 2;
        }

    for (ct = 0; (ct++ < 3) && (!success); )
        {
        success = (biosdisk(DISKFORMAT,drive,head,track,
                   0,NUMSECS,formatinfo)==0);
        success = success && (biosdisk(DISKVERIFY,drive,head,track,
                             1,NUMSECS,0)==0);
        }
    return (success);
    }

***
```

C CODE FOR THE PC

source code, of course

	MS-DOS File Compatibility Package (create, read, & write MS-DOS file systems on non-MS-DOS computers)	$500
	Bluestreak Plus Communications (two ports, programmer's interface, terminal emulation)	$400
NEW!	Turbo Programmer (database application generator; source for library only; specify Turbo C or Microsoft)	$370
	PforC or PforCe++ (COM, database, windows, file, user interface, DOS & CRT)	$345
	CQL Query System (SQL retrievals plus windows)	$325
	GraphiC 4.1 (high-resolution, DISSPLA-style scientific plots in color & hardcopy)	$325
	Barcode Generator (specify Code 39 (alphanumeric), Interleaved 2 of 5 (numeric), or UPC)	$300
	Vmem/C (virtual memory manager; least-recently used pager; dynamic expansion of swap file)	$250
	PC Curses (Aspen, Software, System V compatible, extensive documentation)	$250
	Greenleaf Data Windows (windows, menus, data entry, interactive form design)	$220
	Vitamin C (MacWindows)	$200
	Greenleaf Communications Library (interrupt mode, modem control, XON-XOFF)	$175
	TurboTEX (TRIP certified; HP, PS, dot drivers; CM fonts; LaTEX)	$170
	Essential resident C (TSRify C programs, DOS shared libraries)	$165
	Greenleaf Functions (296 useful C functions, all DOS services)	$160
	Essential C Utility Library (400 useful C functions)	$160
	Essential Communications Library (C functions for RS-232-based communication systems)	$160
	WKS Library Version 2.0 (C program interface to Lotus 1-2-3, dBase, Supercalc 4, Quatro, & Clipper)	$155
	OS/88 (U**x-like operating system, many tools, cross-development from MS-DOS)	$150
	ME Version 2.0 (programmer's editor with C-like macro language by Magma Software; Version 1.31 still $75)	$140
	Turbo G Graphics Library (all popular adapters, hidden line removal)	$135
	CBTree (B+tree ISAM driver, multiple variable-length keys)	$115
	Minix Operating System (U**x-like operating system, includes manual)	$105
	PC/IP (CMU/MIT TCP/IP implementation for PCs)	$100
	B-Tree Library & ISAM Driver (file system utilities by Softfocus)	$100
	The Profiler (program execution profile tool)	$100
	Entelekon C Function Library (screen, graphics, keyboard, string, printer, etc.)	$100
	Entelekon Power Windows (menus, overlays, messages, alarms, file handling, etc.)	$100
	TurboGeometry (library of routines for computational geometry)	$90
	QC88 C compiler (ASM output, small model, no longs, floats or bit fields, 80+ function library)	$90
	Wendin Operating System Construction Kit or PCNX, PCVMS O/S Shells	$80
	C Windows Toolkit (pop-up, pull-down, spreadsheet, CGA/EGA/Hercules)	$80
	JATE Async Terminal Emulator (includes file transfer and menu subsystem)	$80
	MultiDOS Plus (DOS-based multitasking, intertask messaging, semaphores)	$80
	WKS Library Version 1.03 (C program interface to Lotus 1-2-3 program & files)	$80
NEW!	TE Editor Developer's Kit (full screen editor, undo command, multiple windows)	$75
	Professional C Windows (lean & mean window and keyboard handler)	$70
	lp (flexible printer driver; most popular printers supported)	$65
	Quincy (interactive C interpreter)	$60
	EZ_ASM (assembly language macros bridging C and MASM)	$60
	PTree (parse tree management)	$60
NEW!	MicroFirm Toolkit (28 Unixesque utilities for MS-DOS)	$50
NEW!	XT BIOS Kit (roll your own BIOS with this complete set of basic input/output functions for XTs)	$50
	HELP! (pop-up help system builder)	$50
	Multi-User BBS (chat, mail, menus, sysop displays; uses Galacticomm modem card)	$50
	Make (macros, all languages, built-in rules)	$50
	Vector-to-Raster Conversion (stroke letters & Tektronix 4010 codes to bitmaps)	$50
	Coder's Prolog (inference engine for use with C programs)	$45
	Virtual Memory System (least recently used swapping)	$40
	C-Notes (pop-up help for C programmers ... add your own notes)	$40
	Biggerstaff's System Tools (multi-tasking window manager kit)	$40
	PC-XINU (Comer's XINU operating system for PC)	$35
	CLIPS (rule-based expert system generator, Version 4.1)	$35
	Tiny Curses (Berkeley curses package)	$35
	TELE Kernel or TELE Windows (Ken Berry's multi-tasking kernel & window package)	$30
NEW!	SP (spelling checker with dictionary and maintenance tools)	$30
	Clisp (Lisp interpreter with extensive internals documentation)	$30
	Translate Rules to C (YACC-like function generator for rule-based systems)	$30
	6-Pack of Editors (six public domain editors for use, study & hacking)	$30
	Crunch Pack (14 file compression & expansion programs)	$30
NEW!	Pascal Compiler & Interpreter (P-codes, standard Pascal)	$25
	ICON (string and list processing language, Version 7)	$25
	FLEX (fast lexical analyzer generator; new, improved LEX)	$25
	LEX (lexical analyzer generator; an oldie but a goodie)	$25
	Bison & PREP (YACC workalike parser generator & attribute grammar preprocessor)	$25
	AutoTrace (program tracer and memory trasher catcher)	$25
NEW!	Data Handling Utilities in C (data entry, validation & display; specify Turbo C or Microsoft)	$25
	Arrays for C (macro package to ease handling of arrays)	$25
	C Compiler Torture Test (checks a C compiler against K & R)	$20
	Benchmark Package (C compiler, PC hardware, and Unix system)	$20
	TN3270 (remote login to IBM VM/CMS as a 3270 terminal on a 3274 controller)	$20
	A68 (68000 cross-assembler)	$20
	List-Pac (C functions for lists, stacks, and queues)	$20
	XLT Macro Processor (general purpose text translator)	$20
	C/reativity (Eliza-based notetaker)	$15
	Data	
	WordCruncher (text retrieval & document analysis program)	$275
	DNA Sequences (GenBank 52.0 including fast similarity search program)	$150
	Protein Sequences (5,415 sequences, 1,302,966 residuals, with similarity search program)	$60
	Dictionary Words (234,932 words in alphabetical order, no definitions)	$60
	U. S. Cities (names & longitude/latitude of 32,000 U.S. cities and 6,000 state boundary points)	$35
	The World Digitized (100,000 longitude/latitude of world country boundaries)	$30
	KST Fonts (13,200 characters in 139 mixed fonts: specify TEX or bitmap format)	$30
	USNO Floppy Almanac (high-precision moon, sun, planet & star positions)	$20
	NBS Hershey Fonts (1,377 stroke characters in 14 fonts)	$15
	U. S. Map (15,701 points of state boundaries)	$15

The Austin Code Works
11100 Leafwood Lane
Austin, Texas 78750-3409 USA

acw!info@uunet.uu.net

Voice: (512) 258-0785
BBS: (512) 258-8831
FAX: (512) 258-1342

Free surface shipping on prepaid orders **For delivery in Texas add 7%** **MasterCard/VISA**

Reader Service Number 4

SHARE WARE

Help For The Listless

Anthony Barcellos
P.O. Box 2249
Davis, CA 95617-2249
(916) 756-4866

(Here's a TYPE program that's become a LIST, DUMP, filter, and more. After the review of LIST, Tony tells you where to get an index of Micro C.

The MS-DOS TYPE command has one gear: forward. Since TYPE won't scroll backward, the quest for lost text has inspired several deluxe DOS utilities. Vernon Buerg's LIST program is justly the most famous of them.

Computer users speak of LIST as though "excellent" or "wonderful" were part of its name. It's on everyone's short list of indispensable tools. But LIST raises the simple act of screening a file to a high art that most of its users don't appreciate. I thought it was high time to do more than praise LIST in passing. Let's list, as it were, some of its virtues.

How Do I List Thee?

First of all, LIST will list a file (of course). You can scroll right and left in a wide file, as well as the usual up and down. A toggle can turn off the high bit on the extended ASCII set if you are viewing a WordStar document file. Tabs can be expanded. Hexadecimal codes for the ASCII characters can be displayed. LIST can even manage a binary file (like .COM).

Buerg describes the text-processing features of LIST as "filtering." One of his most impressive filtering feats is the elimination of extraneous garbage like control codes (often found in the session-capture files of communications programs).

But Buerg goes one better by giving LIST the power to clean up messy files. That is, not only can you read a file without the interference of "noise" characters and happy faces, you can write to disk a copy that matches the tidy version you've seen on screen.

Shareware authors are getting better organized and more sophisticated these days, but in the past I found LIST indispensable for reading the quirky documents that appeared on distribution disks.

I recall a particularly egregious case in which the author used bare carriage returns (no line feeds) to create overprinting lines. Provided that your printer wouldn't take offense at the file (a big proviso!), the printout would contain bold characters and underlining — things not usually available from ASCII files. Unfortunately, this came at the price of a *much* larger file size (so many lines occurred in duplicate) and limited printer compatibility.

With LIST I was able to create a filtered copy that was nearly a third smaller and printable on nearly any printer. I merely followed Buerg's concise directions in LIST.DOC, the 16-page user's guide to LIST:

While displaying the troublesome file, invoke junk filtering with Alt-J (a mnemonic command!). Press Alt-M to mark the top of the file, press End to jump to the bottom of the file, press Alt-B to mark the bottom, and then hit Alt-D to dump the data to disk. LIST asks you what filename to use and either creates the file or appends the dumped data to the end of the previously existing file by that name.

You don't have to dump the entire file, by the way. Simply use Alt-M and Alt-B to mark the top and bottom of any portion you want. You can hop through a file, picking bits and pieces. If you always use the same filename for the disk dump, LIST will accumulate your data in one place with its append feature.

This is a wonderfully sophisticated piece of work, and Buerg can take just pride in the slick tool he has created. (My appreciation is magnified by my experience in trying to clean the file with the naked carriage returns by my own klutzy programming efforts. LIST did away with all the drudgery once I realized its capabilities.)

Read Your Friendly Manual

LIST's power-user features are no secret. Buerg dutifully sets out the program's many functions in the manual. His collection of "exotic" applications includes file sharing under DOS 3.x, shelling to DOS, and screen saving.

Screen saving? No, Buerg isn't talking about the monitor-blanking utilities that turn off your screen to save the phosphors. He means LIST's peculiar practice of remembering the screen display at the moment it was invoked.

If you exit LIST with the Alt-X command (rather than with Esc or F10), LIST *restores* the screen to its exact appearance before LIST was used! Amusingly, this option leaves no sign of LIST's activity; even the DOS prompt is restored to its original state.

Listing To Port

One advanced application isn't tucked away with the exotic stuff. Rather, the nifty /S option is heralded right on page 1. LIST uses it with redirection and piping. If invoked with the /S option, LIST can grab text that has been redirected or piped to a file or the screen. Buerg's simple example of LIST's piping ability is:

```
dir a: | list /s
```

Instead of going to the screen, as usual, the output of the directory command is channeled to LIST. There you find yourself able to scroll up and down through the directory listing of drive A. That's one good way to improve a DOS command!

Following Buerg's guidelines, I created a batch file that uses Phil Katz's PKXARC in conjunction with LIST to read text buried in an archive file. For the Sacramento PC software library, I created a disk called Volume 0, which contains descriptions of the programs available in the library.

To save space I had to condense the text files into an archive. To help the people who try to use Volume 0 (many of them inexperienced computer users), I arranged to spare them the intricacies of archives by means of LIST's /S feature.

My solution is a batch file called READ.BAT. A command like "read 079" goes hunting in the LIBRARY.ARC file for the text of LIBRARY.079. The /C option of PKXARC extracts the file to the console (screen) instead of to a disk file. The screen output is intercepted by piping it to LIST with the /S feature enabled.

The user can then scroll around and read to his heart's content. When he leaves LIST, the archive is intact, and no additional space has been used by the extracted file.

You *do* need some work space for PKXARC and LIST to do their thing. The batch file won't work on a disk with 0 bytes free. I don't know just how much work space is needed. In my case I did it by trial and error — putting stuff on the disk until it didn't work anymore, then backing off a bit. See Figure 1.

While most people think of LIST as shareware, Vernon Buerg goes his own way, preferring to describe the suggested $15 contribution as a "gift" rather than as a registration fee. It's a gift worth giving.

Vernon D. Buerg
456 Lakeshire Drive
Daly City, CA 94015
(415) 994-2944 (24-hr BBS)

Extended Indexing

In discussing the Letus A-B-C index of articles from computer magazines (July/August 1988), I alluded to the absence of *Micro Cornucopia* from the list of indexed magazines. I'm happy to report that help is available from a different source.

Thomas Brundage of The Logical Connexion has prepared a computerized index for *Micro C.* Mr. Brundage's index is fairly freeform, containing the requisite information on issue, author, article description, and

The DBMS for ***TOUGH*** Programmers

You are an experienced application developer. A specialist. You have worked hard to get here. And you are looking for a DBMS that will maximize your expertise. You know that all the popular DBMSs are designed for the masses — anyone can use them. You know that there are no miracles: a DBMS easy enough for anyone to use ***cannot*** also be the right one for you.

Meet ***The Andsor Collection***: the only DBMS that does not waste your talents. Many find it difficult. Because it was designed for experts. It uses an unusual concept. A combination of built-in operations and procedural language. An environment where the application logic and data are integrated into one unified structure. It baffles novices, but is incredibly powerful in the hands of experts. If you can tame this power, a rich reward awaits you: ***fully customized applications with one tenth the effort***. You deserve it. Because you have worked hard for this expertise.

- □ An interactive environment where you can build and modify an application ***even while it is running***.
- □ Programs ***one tenth*** their size in other systems.
- □ Windows, screens, help, menus, with practically no code at all.
- □ A new way of relating files, can simplify applications by creating relations that are impossible in other systems.
- □ Variable code that changes at run time, lets you perform more than one operation in the same program section.

If you are an average programmer, use the popular DBMSs: they have sold millions, and you need this comfort. ***But if you think you are a tough programmer, accept this challenge***. Write or call for our brochure and ***FREE*** demo disk. It contains two executable, real-life applications, also showing design and programming details. There are comments and help, but no reference manual. Study the applications. Then, if you can understand more than half, congratulate yourself: you are ready to step up to the one DBMS that rewards expertise.

The Andsor Collection™

ANDSOR®
ANDSOR RESEARCH INC.

390 Bay Street, Suite 2000
Toronto, Ontario M5H 2Y2
(416) 245-8073

To order call toll free (U.S. and Canada) **1-800-628-2828 Ext. 535**

$295 **60 DAY MONEY BACK GUARANTEE** Visa, MasterCard, AmEx, Check

Price includes shipping in the U.S. and Canada. Please add $20 for shipping by air to other countries. If you return the software, $15 will be deducted from the refund, to cover shipping and handling.

System requirements: any IBM PC or PS 2 or fully compatible. 320K RAM, one drive or hard disk, monochrome or color

Figure 1 — Text of Read.bat

```
echo off
cls
if x%1 == x goto help
echo Preparing to display Volume %1 information.
echo Use the PgDn, PgUp, and cursor keys to scroll
echo through the information.
echo Press "?" for assistance.
echo Press Esc or F10 when you're through.
pkxarc/c library library.%1 | list/s
goto end
:help
echo The proper way to use the READ command is
echo "READ ###", where ### is a three-digit number
echo from 001 through 099.
:end
```

page number. He uses keywords to tag each item and make it easy to search for information with the DOS FIND filter. Since the index is an ASCII file, you can use any number of tools on it.

The *Micro C* index is available for $10 in either MS-DOS or CP/M format and goes all the way back to issue No. 1. It fits on a single disk. Brundage will update your disk for you provided you send him the *original* diskette, together with sufficient return postage. A short document file on the disk provides further details and limitations.

Thomas Brundage
The Logical Connexion
2523 Phipps Circle, NE
Salem, OR 97305

Growing And Other Pains

Perhaps the shareware industry is maturing. The heavy-duty computer publications like *PC Magazine* and *PC World* now feel constrained to include programs like PC-Write and ProComm when they review software for word processing and communications. *PC World* ran a long piece on shareware in its August 1988 issue.

At the same time, shareware is witnessing a legal wrangle that seems to be breaking new ground. It's reported that Software Enhancement Associates is suing Phil Katz of PKWare for allegedly incorporating features from SEA's ARC in PKWare's PKARC and PKXARC.

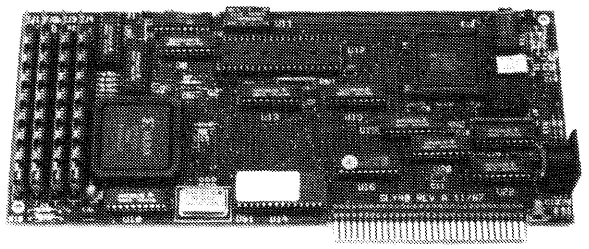
This can hardly be argued on "look and feel" grounds (even if that argument had any merit to speak of) since both SEA's and PKWare's programs just operate from the DOS prompt. What can you do — complain that the command line uses the same letters for different options?

Instead it seems that SEA believes that PKWare is using proprietary portions of ARC's code in PKARC and PKXARC. Is this a credible claim? As a disinterested observer I would have to say that it isn't. Others have already noted that PKWare would severely degrade the performance of its utilities were Katz to borrow code from the sluggish ARC program.

When I ran some simple benchmarks comparing ARC and PKARC (January/February 1988), ARC was left in the dust. It wasn't even a contest. I think SEA should spend more time tweaking its computer code and less time fussing with the civil code.

♦ ♦ ♦

THE SLY40-XT



- The **SLY40-XT** is a 4-¼" by 9-¼" Card Featuring the Full Functionality PC/XT Mother Board
- NEC's 8MHZ V40
- One Megabyte of Zero Wait State RAM
- 64 ROM Space
- Ideal for Tough Industrial Applications

ANNOUNCING THE DISKLESS WORK STATION

- A PC/XT Compatible Network Node
- Enclosure Dimensions are just 10" by 10" by 3-½"
- No Cooling Fans are Required, so the Unit is Perfectly Quiet
- Adds Security to the Company's Computer System
- Dealer & Distributor Inquiries are Welcome

PC and XT Are Trademarks of International Business Machines

Master Card
Visa
Check
Money Order
C.O.D.

SLICER™

Slicer Computers Inc.
3450 Snelling Ave. So.
Minneapolis, MN 55406
612/724-2710
FAX 612/724-0878

Reader Service Number 19

CULTURE CORNER

White Water Rafting: SOG Style

A Very Special Trip

PULL! PULL! PULL!

There's been a rumor floating around that computer technical journals have sold out to advertisers. Let me say allegorically, right here and now and for well into the future, it's not true.

For instance, this is the special SOG white water raft trip we set up just for advertisers. Note the whip, note the buckets of water, note the eyes. Are they having fun yet?

ON YOUR OWN

The Future Of "On Your Own"

A Change In The Guard

By Gary L. Scott
Decision Technology
P.O. Box 5040
Aloha, OR 97006
(503) 642-4196

Okay ... I talk to you, I plead with you. It does no good. You think you can just go off and start your own business. Be a big shot.

You're no doubt the same person who leaps out of planes without checking to see if the pool's been filled and jumps off high dives without a parachute. Boy, will Gary straighten you out.

I have taken over responsibilities for this column from Dave and the gang at *Micro C.* This should give Dave extra time to write those looo...oong editorials he is so famous for. As with every changing of the guard, I plan to make some changes to the format of this column. My plan is to divide the column into two parts, a description of an existing business, and the discussion of a topic that will be helpful to the readers starting or running a business.

Taking over the column was easy. Keeping the ideas flowing is not so easy. I would like nothing better than to write about *Micro C* readers, your successes and your not-so-successes. If you have a business that you have started and would like it featured in this column, please contact me at the above address or via *Micro C.* I would also welcome letters with questions or suggestions regarding the column.

I am particularly interested in talking to any of you who are selling high tech consulting services, even if you only do it part time. I would like to do a column on consulting so I need to talk to as many of you as I can. Also, *Micro C*'s next theme issues will deal with robotics, CAD/CAM, and real-time/process control. If you are currently working in any of these areas, give me a call between 6 and 10 p.m., Pacific Time.

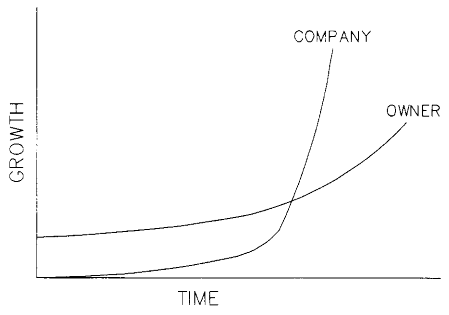
Death Of A Company

In my experience, the thing that kills more young companies than anything else is lack of education, and I don't mean advanced degrees. If we look at Figure 1, there are two lines plotted on the graph. The first is growth of the owner and the second is growth of sales. Companies normally grow and prosper as long as the growth of the owner stays ahead of the growth of the company. When the company grows faster than the owner, chaos is almost a certainty.

Growth

It is a time-warn cliché that a business never stands still, it is either growing or it is shrinking. There are two forms of growth and each has its own problems. The two types are slow growth and rapid growth. Any businessman worth his salt will tell you what he wants and fears the most is fast growth.

Figure 1 — Company Owner Growth.



At first glance it may seem that fast growth is what you should strive for. But is it? Everyone knows who Apple Computer and Compaq are, but how many people remember Osborne computer, Victor PC, Digital Research, The Pet Rock, or any of the other many examples of businesses that took off like a rocket only to crash?

Remember when Digital Research was on the top of the heap and Microsoft was a struggling company? What differentiates these companies is that Apple, Compaq and Microsoft were able to make the mid-course corrections necessary to continue their growth.

How can you keep this from happening to your enterprise? Specialized education is often the answer. There are a number of alternatives for dealing with growth and that's what this column is about.

Books

Probably the best of the non-technical books on starting and growing a business is *Growing a Business,* by Paul Hawkins. This was a companion book for a PBS series on starting a business. If you can find video tapes of the series, get them, they are well worth watching.

If your background steers you toward more technical tomes, the best two books are by Michael Porter, *Competitive Advantage* and *Competitive Strategy.* Porter does a very good job of describing the inner-workings of the business machine, but these are definitely not books for a beginner. They are hard reading even for a business school graduate.

If you are considering starting your own business but haven't made the move yet, check out three workbooks: *Venture Feasibility Workbook,* by Robert Ronstadt, *Your Business Plan* and *Your Marketing Plan,* by the Oregon SBDC Network.

Both the *Venture Feasibility Workbook* and *Your Business Plan* are designed to help you develop a solid business plan. *Your Marketing Plan* was written to augment and expand the marketing section of *Your Business Plan.*

If you have written business plans in the past, you will probably want to purchase a copy of *Venture Feasibility Workbook* first. It takes a unique approach to the traditional business plan.

However, whichever workbook you begin with, keep in mind one thing. If you cannot answer the questions in these books, you are not ready to start your company.

A good book on the financial aspects of starting and growing a business is *Entrepreneurial Finance,* by Robert Ronstadt. This book has a heavy tie-in with the software package, Ronstadt's Financials (discussed later in this column), but is also a very good standalone tool. *Entrepreneurial Finance* has the best discussion I've seen of the financial problems facing a new business. Ronstadt does a very good job of explaining this very necessary and important part of new business planning.

I have sent a list of books for posting to the Micro C bulletin board. If you are interested in reading more on starting a business, get a copy of this bibliography and a library card. (You could start a small business for less than all these books will cost.)

Information needed to order the workbooks is as follows:

Venture Feasibility Workbook
by Robert Ronstadt
Lord Publishing
(508) 651-9955; $19.95

Your Business Plan &
Your Marketing Plan
by the Oregon SBDC Network
Portland Community College
Small Business Development Center
(503) 273-2828; $10 each

Software

There are a number of programs on the market that claim to help you write your business plan or run your business. After looking at a number of them, my feeling is that most of them are not worth the money. There is one notable exception.

If you are planning on starting a business in a big way, you might want to consider purchasing a copy of Ronstadt's Financials. Though it costs $500, it's the best package that I've seen for developing financial scenarios.

Small Business Administration

The SBA has a program that uses retired executives to consult with businesses. The few times that I have been exposed to this program have led me to the opinion that most of these folks come from "large" businesses and do not have much relevant information to impart to the small businessman.

Even though my exposure to this program has been less than terrific, this is still a resource to be considered. There are many very capable individuals out there who are volunteering their time through the SBA program.

Small Business Development Centers

The SBDC network is usually attached to the state college system. In Oregon they fall under the community colleges; in other states they might be attached to four-year institutions. SBDCs are normally staffed by instructors and/or people with specialized business experience.

The big advantage to the SBDC network (like the SBA) is that you can normally find someone to talk with without shelling out money. Yes, they try to sign you up for their classes and try to sell you their publications, but everything is reasonably priced. A few hours spent with a SBDC counselor can save you many days of frustration and, more important, a lot of money.

As a side note, Dave & *Micro C* are graduates of the Central Oregon Community College SBDC. If you want an (un)biased opinion, call Dave.

Editor's note: The SBDC in Bend is great. When Micro C was in its rapid growth phase (5 years ago), I was totally overwhelmed. It turned out that the college was just setting up the SBDC here when I *called them (I was one of their first cases). Fortunately, I called soon enough. Many of the small businesses they have contact with are already in their death throes. The only thing left for those is last rites.*

The SBDC's classes on starting small businesses are excellent. Most of the folks have come out of those sessions realizing their dreams would be a waste of time and money. Others have come out with an even stronger belief that they'll make it because they have a better feeling for what they'll be facing and what they'll need to cope.

The SBDC will also give you an opportunity to help others. I've donated many hours talking to local folks who dreamed of starting their own publications or marketing their own computer products. In the case of one publication, the person had no idea where to get articles, what printing cost, or where to find subscribers. (We're ALL looking for subscribers.) One computer product was a very expensive, very limited, very unfinished, schematic capture program which would (eventually) run on a no-longer manufactured (or supported) computer.

Three years ago I met with a group which had invested in an Oregon hard drive manufacturing startup. They came over (from Medford, if I remember correctly) to ask me if they should pony up another $2,000,000 or lose the $5,000,000 they'd already invested. They proudly showed me a sample of the 10 meg drive designed by their Bay Area guru.

I asked them if they could sell the drive for $200 a copy and make a profit, figuring warranty, customer support, etc. From the shock on their faces, I knew the answer. I recommended they save their $2,000,000.

Later, I heard they'd put up the money, but the company folded shortly thereafter. The Seagate 225 had killed them.

Background

It seems like almost every time a new column is started there is the obligatory discussion of the writer's background. To that end I have included the following thumbnail sketch.

I have worked in the high tech industry as a software engineer for over fifteen years. During most of that time I have consulted on the side as a way to pay for my expensive high tech toys. My clients have ranged from no-tech to super-tech.

I've spent the last two years completing my Masters in Management at a local college. During that time I spent a year working as a sales manager for a small computer VAR, and am currently employed as a business manager for a small no-tech consulting company.

My time is currently split between consulting on PC-based projects and independent management consulting for the Center for Entrepreneurial Ventures (CEV) in Portland, Oregon. The CEV is a small publicly-funded training program for individuals who are either currently running a high growth business or are looking for nonfinancial help in starting a new company.

♦ ♦ ♦

C_talk™ The Practical Union of C and Smalltalk

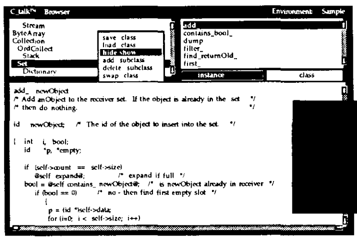
Add a new dimension to your C compiler.

From C:
- Ease of application delivery – portability
- Performance – speed and efficiency of C
- Familiarity of C – use all your existing C code

From Smalltalk:
- Data abstraction – data hiding/encapsulation
- Full object inheritance
- Polymorphism – message sending with dynamic binding

Boost Your Productivity!* *C_talk's practical approach to object-oriented programming in C allows you to realize substantial productivity gains using these tools:

- C_talk's *Browser* – a powerful Smalltalk-like browser for building software objects
- An automatic *Make* utility – for building applications
- A *Preprocessor* – for converting objects into C source code.
- A set of *Foundation Classes* – to use as basic building blocks.



```
add_ newObject
/* Add anObject to the receiver set. If the object is already in the set   */
/* then do nothing.                                                        */

id   newObject;   /* The id of the object to insert into the set.   */

{  int   i, bool;
   id    *p, *empty;

   if (self->count  ==  self->size)
      @self expand@;             /* expand if full */
   bool = @self contains_ newObject@;  /* is newObject already in receiver */
      if (bool == 0)        /* no - then find first empty slot */
         {
          p = (id *)self->data;
          for (i=0; i < self->size; i++)
```

$149⁹⁵

Why C_talk?

C_talk has been proven successful in delivering several large-scale systems in demanding realtime environments. It's concise, easy to learn and use. It is programming in C (not a new language), while adhering to the Smalltalk paradigm.

C_talk is the practical, and affordable, union.

C_talk is designed to operate with MSDOS on IBM or compatible computers. At least 512K of memory, a hard disk and mouse are recommended.

Order today!

Call or write:
CNS, Inc.
Software Products Dept.
7090 Shady Oak Rd.
Eden Prairie, MN 55344
Tel: (612) 944-0170
Fax: (612) 944-0923

Add for shipping $5 US, $25 Int'l.
(30-day money-back guarantee)

CNS®

...providing and advancing object-oriented methodology.

C_talk is a trademark of CNS

CNS is a registered trademark of CNS, Inc.

Reader Service Number 118

Around the Bend

Continued from page 4

planes ahead of me. Hoping the one behind me hadn't gotten lost. (He had.) We buzzed three airports, all prearranged. No one noticed.

I spotted a hawk, maybe 20 feet above me. He wheeled sharply left and disappeared.

So much for yesterday, now I have to deal with a damp, surprisingly noisy dawn.

My air mattress is sagging. I wouldn't have noticed if that duster pilot had stayed in bed. But he didn't. So I notice. And now I, a hard-core night person, am writing in a tent, before sunrise, in a strange and lumpy place.

Boy, my fingers get stiff in this cold air. Wait a moment while I tuck them into the sleeping bag ...

There, that's better. The door of my mountain tent is open, it gives me just enough light to see the Kaypro 2000 screen. Really lets in the cold, though.

Very still outside, the air hasn't awakened yet.

My fellow travelers? I don't know. There are all types; except for a couple they're pretty much retired. Most have lots of time for piddling with airplanes, a few have money. Some of the craft are owned by as many as 13 people, and I can imagine 13 people standing in a circle drawing straws to see who gets the plane for this tour.

Most are pretty simple: 65-horse Continental or Franklin (whatever was around 40 or 50 years ago). Tiny wood or metal prop. A wooden or tubular frame covered with cotton and varnish (lots of varnish). No radio. Few instruments. Conventional landing gear with tiny tail wheels. Some don't even have compasses. (We each received a map showing the week's route. Now I understand why it's a highway map.)

Shucks, the crop duster is back. Landing on the taxi way, he sidles up to his giant chemical vat and refills. A chemical smell slowly envelopes my tent. With a second roar, he's off again. Coughing has almost completely drowned out the snoring.

The snoring's coming from the Jack tent. I had expected better from pilots, but there was a smoking and drinking (never go anywhere without their Jack) crowd. Smoking is deadly, especially above 10,000 feet. They can't breath.

My hands are getting really cold, think I'll just shut down.

Sunday 4:44 p.m., Roseburg

I'd assumed we were flying low because of the clouds and because we were tiptoeing through the Portland Control Zone. But the weather has cleared a bit, Portland is well behind us, and we're still hugging the deck. Close enough to read highway signs, close enough to race, nose to nose, with unsuspecting cars, close enough to watch kids play squareball on small town streets.

I'm on edge, traveling through narrow valleys. Rounded hills which look insignificant from 10,000 feet now rise far above me.

After the rude wake-up, this morning was eventful. Low scud hid the hills as we crawled out of tents and bags, so we aborted our planned hop to Florence.

Florence is on the Oregon coast, which means ducking through twisty little valleys to reach the ocean. (Then you face the coastal fogs.)

Instead we headed south, following, as long as we could, the broad Willamette valley. A few minutes after takeoff I

E=M C² AZTEC

Genius Begins With A Great Idea...

FREE 2 DAY DELIVERY

Aztec C86 4.1
New PC/MS-DOS
CP/M-86 • ROM

Superior performance, a powerful new array of features and utilities, and pricing that is unmatched make the new Aztec C86 the first choice of serious software developers.

Aztec C86-p....................$199
• optimized C with near, far, huge, small, and large memory - Inline assembler - Inline 8087/80287 - ANSI support - Fast Float (32 bit) - optimization options • Manx Aztec 8086/80x86 macro assembler •Aztec overlay linker (large/small model) • source level debugger • object librarian • 3.x file sharing & locking • comprehensive libraries of UNIX, DOS, Screen, Graphics, and special run time routines.

Aztec C86-d....................$299
• includes all of Aztec C86-p • Unix utilities make, diff,grep • vi editor • 6+ memory models • Profiler.

Aztec C86-c....................$499
• includes all of Aztec C86-d • Source for library routines • ROM Support • CP/M-86 support • One year of updates.

Third Party Software

A large array of support software is available for Aztec C86. Essential Graphics • C Essentials • C Utility Library • Greenleaf Com. • Greenleaf General • Halo • Panel • PC-lint • PforCe • Pre-C • Windows for C • Windows for Data * C terp • db_Vista • Phact • Plink86Plus • C-tree.

C' Prime
PC/MS-DOS • Macintosh
Apple II • TRS-80 • CP/M

These C development systems are unbeatable for the price. They are earlier versions of Aztec C that originally sold for as much as $500. Each system includes C compiler, assembler, linker, librarian, UNIX routines, and more.. Special discounts are available for use as course material.

C' Prime $75

Aztec ROM Systems
6502/65C02 • 8080/Z80
8086/80x86 • 680x0

An IBM or Macintosh is not only a less expensive way to develop ROM code, it's better. Targets include the 6502/65C02, 8080/Z80, 8086/80x86, and 680x0.

Aztec C has an excellent reputation for producing compact high performance code. Our systems for under $1,000 outperform systems priced at over $10,000.

Initial Host Plus Target...$ 750
Additional Targets............$ 500
ROM Support Package....$ 500

Vax, Sun, PDP-11 ROM HOSTS

Call for information on Vax, PDP-11, Sun and other host environments.

Cross Development

Most Aztec C systems are available as cross development systems. Hosts include: PC/MS-DOS, Macintosh, CP/M, Vax, PDP-11, Sun, and others. Call for information and pricing.

CP/M • 8080/Z80 ROM

C compiler, 8080/Z80 assembler, linker, librarian, UNIX libraries, and specialized utilities.

Aztec C II-c CP/M & ROM....$349
Aztec C II-d CP/M.................$199

How To Become A User

To become an Aztec C user call 800-221-0440. From NJ or international locations call 201-542-2121. Telex: 4995812 or FAX: 201-542-8386. C.O.D., VISA, Master Card, American Express, wire (domestic and international), and terms are available. One and two day delivery available for all domestic and most international destinations.

Aztec Systems bought directly from Manx have a 30 day satisfaction guarantee. Most systems are upgradable by paying the difference in price plus $10. Site licenses, OEM, educational, and multiple copy discounts are available.

MANX

To order or for information call today.

1-800-221-0440

In NJ or international call (201) 542-2121
TELEX 4995812

Manx Software Systems
One Industrial Way
Eatontown, NJ 07724

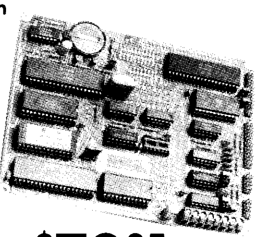
Reader Service Number 17

THE NEW 65/9028 VT
ANSI VIDEO TERMINAL BOARD!
★ FROM LINGER ENTERPRISES ★

A second generation, low cost, high performance, mini sized, single board for making your own RS232 Video Terminal. This highly versatile board can be used as a stand alone video terminal, or without a keyboard, as a video console. VT100, VT52 Compatible.

FEATURES: **MICRO SIZE!**

- ★ Uses the new CRT9128 Video Controller driven by a 6502A CPU
- ★ On-Screen Non-Volatile Configuration
- ★ 10 Terminal Modes: ANSI, H19, ADM-5, WYSE 50, TVI-920, KT-7, HAZ-1500, ADDS 60, QUME-101, and Datapoint 8200
- ★ Supports IBM PC/XT, and Parallel ASCII Keyboards
- ★ Supports standard 15.75 kHz (Horiz.)
- ★ Composite or Split Video (50/60 Hz)
- ★ 25 X 80 Format with Non-Scrolling User Row
- ★ Jump or Smooth Scroll
- ★ RS-232 at 16 Baud Rates from 50 to 19,200
- ★ On Board Printer Port
- ★ Wide and Thin Line Graphics
- ★ Normal and Reverse Screen Attributes
- ★ Cumulative Character Attributes: De-Inten, Reverse, Underline and Blank
- ★ 10 Programmable Function Keys and Answerback message
- ★ 5 X 8 Character Matrix or 7 X 9 for IBM Monitors
- ★ Mini Size: 6.5 X 5 inches
- ★ Low Power: 5VDC @ .7A, ± 12VDC @ 20mA.



$79.95 FULL KIT
w/100 Page Manual
ADD $40 FOR A&T

OPTIONAL EPROM FOR PC/XT STYLE SERIAL KEYBOARD: $15

SOURCE DISKETTE: PC/XT FORMAT 5¼ IN. $15

Digital Research Computers

P.O. BOX 381450 • DUNCANVILLE, TX 75138 • (214) 225-2309

Call or write for a free catalog on Z-80 or 6809 Single Board Computers, SS-50 Boards, and other S-100 products.

TERMS: Add $3.00 postage. We pay balance. Orders under $15 add 75¢ handling. No C.O.D. We accept Visa and MasterCard. Texas Res. add 6-1/4% Tax. Foreign orders (except Canada) add 20% P & H. Orders over $50 add 85¢ for insurance.

Reader Service Number 32

dp_MAX

dBase III Tools in Turbo Pascal 4.0

Complete Support for dBase III files.
DBF, NDX, DBT file & record Access.
Fully Compatible dBase III B+Tree ISAM.
Library of 100+ functions in TP4 Unit.
Allows 250+ files & indexes open at once.
LRU file caching, round robin file manager.

$$\text{dp\_MAX} = \int_{-\infty}^{+\infty} \frac{\text{dBASE III +}}{\text{Turbo Pascal 4.0}}$$

Max Software Consultants, Inc.
4101 Greenmount Avenue
Baltimore, MD 21218
(301)-323-5996

$149

Reader Service Number 103

Around the Bend

Continued from page 69

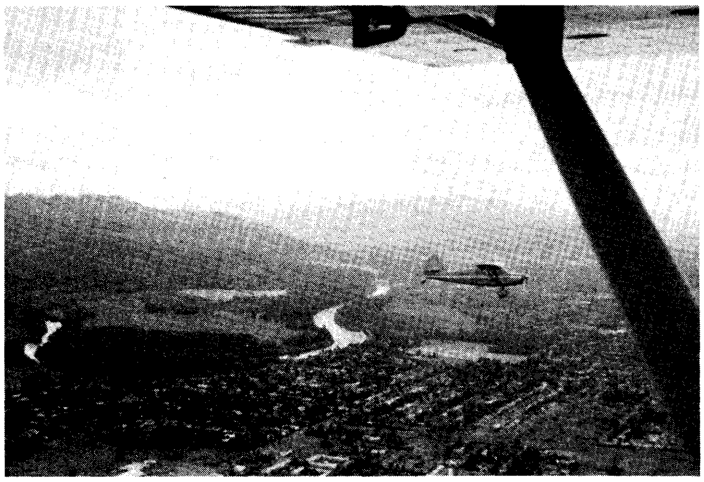
noticed my generator wasn't working. After kicking myself for not paying more attention during run-up, I shut off running lights, strobe, radio, everything that used electricity (the spark plugs have their own power) and grabbed the map. The largest nearby airport was Salem; I waited until it showed up off my right wing and turned on the radio.

"Salem Tower, Stinson 8077K, five miles east, with an electrical problem. Is there a mechanic on the field?"

"77K wait one.

"77K, Roger on the mechanic. Active runway is 31, you're cleared to land."

The mechanic found a tripped circuit breaker and I was on my way again. (Hey, I help people with their computer problems all the time, and I didn't suspect a silly circuit breaker? Cost me $10, too! Ah well, it's vacation.)



Flying in Formation with another Stinson.

The Cottage Grove Experimental Aircraft Association was holding its summer barbeque, so (by pure coincidence) we (22 planes) dropped in for lunch. Made very quick work of their hamburgers.

It was in Cottage Grove we learned that Florence had been waiting for us. They'd had 150 people at the airport, complete with the Florence High School Band. (A real band, no one's ever met me with a band before.) We gathered in the Cottage Grove airport office to share the news and kick ourselves for not sending out scouts (planes with radios) to see if the coast was clear (or clear enough).

The trip's been so exciting, we'd have risked life and limb to hear that band. Gives you some idea.

As I enter this, I'm sitting next to the Stinson on hot gravel-covered blacktop. We're in Roseburg, our overnight stop, and I'd better get off this computer and set up the tent. The red sun isn't going to be red forever, and nights here can be thunderstormy.

Monday 8 p.m., Grants Pass

First day of work I've missed in a long time. Feels good to be away. Almost forgot about the magazine, *Micro* ... ???

It's a lazy afternoon, clear, sunny, warm, north wind. I'm sitting on my bedroll, leaning against the shady side of the Stinson. Gusts are strong enough to shake the old bird occasionally. Someone said 20 knots. Tried to catch a nap inside the plane, but

Around the Bend

Continued from page 70

the sun heats things up too much when the doors are closed. When they're open, anything loose takes flight lessons.

Local flying aficionados put on a picnic for us at a physician's house. It's obvious that planes make wonderful food-fellows. At least here. We're talking roast ham and turkey, uncountable salads and vegies, beer, wine, pop, and desserts. (Let me tell you about the desserts.) Stuffed ourselves for two solid (stolid) hours.

Tonight's feast was a real improvement over last night when our fare resembled untanned leather. The locals called it pizza.

Am carrying about 60 pounds for a couple who are traveling in an ancient two-place Interstate. (I guess the name comes from the type of road they follow.) They can't haul anything more than themselves. (Themselves being significant.)

Boy, do sixty additional pounds fill up the back seat. I wonder what's happening to my weight and balance, but the old bird doesn't seem to notice. I burned about 6 gallons during the half-hour 50-mile flight from Roseburg to here (Grants Pass).

Grants Pass is drier than anything we've seen so far and it's very hilly, much like Northern California. More afternoon winds, glad we arrived early.

Tuesday Evening, Montague, Calif.

It's blowing like a banshee here, wind sock as straight as starched slacks. Gusting to 30, really shaking the plane despite the ropes. A local said there's a chance of an afternoon thunderstorm if the wind dies down.

Haven't seen a windier place than this. People bend down and hunch their shoulders, even when they're inside.

Air was quite rough coming in so it was a real relief to get the Stinson on the ground. For a lark, I landed on the hay field they said was their alternate runway. (Guess that makes them a real airport, having an alternate hay field.) It was rougher and shorter than it appeared from the air.

The members of the group who watched me come in figured I'd just missed the runway. They might have been right.

This morning, we took off from Grants Pass and flew about 60 miles into the mountains to a small untended airstrip near the logging community of Prospect. The strip was an uphill/downhill swath in the woods with some very sturdy looking firs at each end.

The town, an easy half-mile walk from the strip, sported a single small diner. As we nonchalantly wandered in the front door, the waitress grabbed her apron, both menus, and announced that their grill was only so by so. However, she came up with 20 lumberjack-sized breakfasts in less than an hour.

One plane, a Piper, had magneto trouble at Prospect, right mag quit. (This is serious. If both right and left quit, the engine quits.) So he let his passenger ride in another plane while he flew to Medford. I followed him in the Stinson in case of further trouble.

Ah, well, back to the present.

Having a laptop along has been great, it needs a lit screen so I can write in the tent at night without fussing with a candle — burned my fingers twice. But the computer's better than paper and I can recharge the little beastie while I'm flying.

Looking around, I'm surprised. The Montague airport is gamey as airports go, tin hangars, white peeling buildings, grass so dead it crackles underfoot. Really feels like a ghost town.

But the people who work here, and the flying buffs who

C++

FULL COMPILER – not a preprocessor

- Complete software-development system
- Selectable AT&T C++, ANSI C, K&R C
- Based on proven Oregon Software technology
- Generates extremely fast, compact code
- Comprehensive error checking
- Backed by responsive support engineers
- Strongly typed language

DATA ABSTRACTION FACILITY (classes)

- Operator overloading
- Information hiding/sharing
- Constructors/destructors

✓ **YES!** **I want top performance at an affordable price**

To order, or for more information, call 1-800-874-8501
6915 SW MACADAM, SUITE 200, PORTLAND, OR 97219

OREGON SOFTWARE
Professional Products for Software Development

Reader Service Number 85

HANDS-ON PC REPAIR TRAINING

NAC offers the highest quality advanced training available for PC technicians. The course and its documentation focus on the IBM personal computer lines, from PC to PS/2, peripheral devices, and compatibles such as Compaq and AT&T. Network maintenance training is also available.

- Diagnose more quickly and accurately, complete repairs faster, and reduce costs.
- Discover better diagnostics, parts sources, and new trouble-shooting tools and test units.
- Toll-free technical help and parts-locator line, free technical and vendor update service.
- Used by GTE, ITT, Rockwell, Xerox, SoCal Edison, regional Bell companies, the U.S. Dept. of Commerce, and many others.

Our five-day PC Maintenance Course is offered in California, New York, Washington, Atlanta, Chicago, St. Louis, Dallas, Seattle, and other major cities. Call without obligation to get a course outline and dates for the class locations most convenient to you.

National Advancement Corporation
2730-J South Harbor
Santa Ana, CA 92704
(714) 754-7110
Computer Hardware Training Specialists

Reader Service Number 59

Continued from page 71

hang out here, go out of their way to be friendly. Plus there's a shower, a real, hot, wet, free, work-up-a-warm-lather, shower. Haven't gotten in line yet but will definitely dig out the soap and before the evening's over. (First shower in four days.)

Don't know where I'll pitch the tent. Most of the ground is gravelly and covered with stickerburrs. But I don't want to spend the night folded up inside the plane, and I don't want to get blown away. Shucks, was hoping to throw the sleeping bag under the wing, just once.

We've been joined by a home-built, sort of a flying wing thing with its tail out front and tiny 2-cycle motor in the back. (Maybe they just put the prop on backwards.) Goes 85, stalls at 50, sounds like a sewing machine. The huge wing appears very hard to control in turbulence.

We're trapped if the weather gets nasty, but that's the way it is with this kind of flying. I guess I shouldn't worry about it, I'm on vacation.

It's fun sitting here, writing for the joy of it. Writing to get down the feelings. To get down the thoughts I don't want to lose. It's still fresh, this touring around. It's an adventure.

Oops, last person's out of the shower, guess I'll fold up this computer and grab my washcloth.

Tomorrow I'll break out of the pack, climb to 10,000 feet, and make the 2-hour dash to back to the office to see if everything's OK.

As you probably noticed last issue, Carol will soon tell me that last issue's editorial is too short. (This isn't anticipatory journalism, this is weird.)

DISK FORMAT CONVERSION

XenoCopy-PC

$79.95 + $5.00 S/H
Sales Tax if CA.
UPS COD

PC-DOS program lets your PC

READ / WRITE /
FORMAT / DUPLICATE
Disks from over 300 other micros

Upgrades availabe from previous versions for only $25.00 Call for Authorization

To Order Contact:

XENOSOFT™

2210 SIXTH ST. BERKELEY, CA. 94710
(415) 644-9366

Reader Service Number 39

John Jones

If you have a copy of *Micro C* Issue #1, you know that John Jones wrote a short piece on his disk formatter for the Big Board (complete with listing). He showed up again in #2 and, well, has been very regular since.

After seven years, he's called it quits.

Thanks John. I've appreciated your help a lot. A whole lot.

Looking Ahead

You were all great when we asked for database information. So we're asking for more ideas.

Here are the (tentative) themes for the next year:

Computer Aided Design — Jan/Feb

Bruce Eckel, Sam Azer, and a whole group of assistants are working on comparative reviews of CAD packages, but I'd like someone to come up with a complete tale of joy and woe as he did a complete project using CAD for everything from schematic capture to film for the PC board. (A blow-by-blow account similar to my ongoing saga with desktop publishing.)

Tools — Mar/Apr

Okay, so you don't qualify as a robot. You know what tools are. We're talking libraries, debuggers, logic simulators, program generators ... You know, tools.

If you write tools, or if you use or test them extensively or know what tool creators should be working on, then shout.

Robotics — May/June

My first flash when someone says robot is Heath's early do-it-yourself rug rat. Several engineers in my group at Tektronix used one of these wire-controlled kludges to clear the area of people from the manuals group. (It worked.)

Robots are, however, much more than novelties. Twenty years ago we knew that humanoid robots would soon be doing the manual labor for society. Well, I haven't seen a humanoid in Bend lately (they've probably all moved to the East Coast), but that doesn't mean we don't have robotics.

If you're working on or with a computer that can see, speak, drill, shape, weld, move ... get in touch. (Either you, or the computer.) I'm looking for theory and practice. (The Marines are looking for a few good humanoids).

Handicapped Systems — July/Aug

Of course, most systems are handicapped. But I'm referring to ways to make computers more available to the sensory impaired and to those who have trouble moving limbs.

I've been thinking about this lately, and I know that as voice synthesis gets better and cheaper, it'll become standard in all systems, just as graphics have replaced character displays. That means I could have my X-16 read back my editorials. (Faster and less addicting than sleeping tablets.)

I/O Systems — Sept/Oct

These are controllers, monitors, alarm systems, atmospheric testers, and so on. The possible topics here include: a look at controller processors (I/O features instead of MIPs), AtoD, DtoA, converting temperature, pressure, humidity, stress, position, to voltage or current. Polling remote sensors ...

Plus there's the software end of this. How about the problems of developing and testing programs? Building embedded systems? The PC is a good development system, how about using it as a hardware platform, too?

UNIX — Nov/Dec

I'm looking for some UNIX pieces to run before the Nov/Dec 1989 issue but I'd like a real blow-out UNIX issue

Continued from page 72

for Christmas. With the popularity of 386 systems and the problems OS-2 is having, UNIX might just take off. (We'll shell-shock 'em.)

So:

If you're heavily into any of these areas, have other suggestions for articles, or whatever, call Larry, Cary, or me at 503-382-8048, or write (P.O. Box 223, Bend, Oregon 97709), or leave a message for the SYSOP on the Micro C RBBS (503-382-7643, 300,1200,2400).

SOG VII

We had quite a group of women at SOG VII. Two of the stars were Deborah Norling and Christy Quinn of Grassroots Computing. Grassroots Computing puts together hardware and software systems for the handicapped. Though Debee is blind, she was an active participant in everything from white water rafting and barbeque to doing two impromptu presentations. During two 11 p.m. sessions, she demonstrated how she writes software and reads publications using only her laptop and a voice synthesizer.

Debee Norling and Christy Quinn at SOG.

After SOG I called Grassroots and talked to Christy. After meeting Debee, I was really excited about the role computers could play in the lives of the country's handicapped. Christy, a professional counselor, painted another picture.

"Eighty-five percent of the blind are unemployed — on welfare. They get discounts on everything, special privileges, and they're used to having people do everything for them. They're the receivers. Getting something for nothing. We set these people up with computers and they wind up in the closet (I think she means the computers). Every once in a while, we make a good mistake and someone takes it and runs with it, just like Debee did.

"There's a cerebral palsy guy who's really blossomed; he lives on it, wrote his first letter to his folks after getting his system. For two years another fellow just formatted disks, then he blossomed.

"SOG was really exciting for Debee. After a couple days she decided she really fit in. Had lunch with Walter Bright (author of Zortech's C++); they had a great discussion and she came away feeling very good about herself. She felt professional, an equal, not just a poor little blind girl.

"But again, Debee is one of a handful of blind people in the U.S. who are really trying to contribute."

Debee's demonstrations of her system were a real hit at SOG. We've got her scheduled for a formal presentation of computing for the visually impaired at SOG VIII. By the way, we're scheduling SOG VIII for July 13-16. Mark your calendars.

If you want to get in touch with Debee or Christy, they're available at:

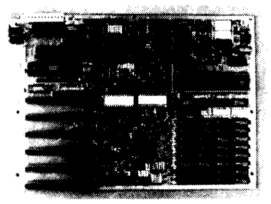
Deborah Norling & Christy Quinn
Grassroots Computing
PO Box 460
Berkeley, CA 94701
(415) 644-1855
Compuserv ID: 72236,2655

Missing Author

A couple of months ago, we received a really interesting description of the language Tenne-C. It's a substandard language, replete with expletives. For instance, the loop and conditional constructs include:

```
Hauloff and Do
Fer, til loop
Whol, longasyerattit
Yehbut, nowait
```

68000 SINGLE BOARD COMPUTER



- 512/1024K DRAM
- 4 RS-232 SERIAL PORTS
- FLOPPY DISK CONTROLLER
- REAL TIME CLOCK

BASIC KIT(8 MHZ) - BOARD, MICROPROCESSOR, HUMBUG MONITOR/ BASIC IN ROM, 4K SRAM, 2 SERIAL PORTS $200

PERIPHERAL TECHNOLOGY PROVIDES ACCESSORIES TO BUILD COMPLETE SYSTEMS!

PACKAGE DEAL - COMPLETE KIT WITH 10 MHZ MICROPROCESSOR, SK*DOS OPERATING SYSTEM, 512K DRAM $575

SYSTEM BOARD(12MHZ) - ASSEMBLED/TESTED, 1MEG RAM, 6 PC/XT PERIPHERAL PORTS, SK*DOS $899

COMPLETE INFORMATION AVAILABLE UPON REQUEST

PERIPHERAL TECHNOLOGY
1480 TERRELL MILL ROAD SUITE 870
MARIETTA, GA 30067 404/984-0742

COD/MASTER CARD/VISA/CHECK
SK*DOS IS A TRADEMARK OF STAR-K SOFTWARE SYSTEMS CORP.

Reader Service Number 103

Around the Bend

Continued from page 73

It's great Culture Corner material and it's signed by Andrew B. Peed of AT&T Bell Laboratories. We haven't found Andrew, yet. Another clue: A note at the end states it originated at the Micro Message Service RBBS 919-779-6674. (However, the SYSOP didn't know how he got it.)

I'd love to find the original author and get permission to Hauloff and Do this'n.

Disk Technician

I mentioned Disk Technician in my last hard drive article and then forgot to tell you where to find it. It's available from:

Prime Solutions
1940 Garnet Ave.
San Diego, CA 92109
(619) 274 5000

Dave Thompson, Larry Fogg and Gary Entsminger hold an impromptu SOG meeting.

Pricing Your Product

Earlier today I was on the phone with a prospective advertiser. At the beginning of the conversation, I mentioned that he would not do well in *Micro C* until he reduced his price. (Over $300 for a library with source.)

While we were discussing his ad, he mentioned he was getting an impressive number of bingo inquiries (circle the number ...) from his ad in *Computer Language*.

"They're interested but they're not purchasing," I guessed.

He agreed.

He said he had called some of the prospects to see why they weren't ordering. Only one person mentioned the price. Everyone else said:

"We're already in the project cycle."

"Don't have a use for it yet."

"It isn't quite what I was looking for."

He also said the people who were using it had said the real value of the package was far more than the price. And he added that they've got a 700 page manual which costs them $50 a copy.

"Put yourself in their shoes," I countered. "From what you see in the flyer or the ad, would you order the library for $325? Would you order it for $99?"

"That's a no-brainer. Of course I'd jump on it for $99."

Which makes my point. People always weigh price against the perceived value. (Forget about real value, these folks don't have the product yet.) If the perceived value wins they buy it. If not, they don't. The higher the ratio of perceived value to price, the larger the audience.

So: everyone he talked to told him the price was too high. He just didn't realize it.

Finally. There's a lot to be said for just getting copies out there. It gets your product into people's hands so they can use it, get to know it, tell their friends about it, purchase upgrades. (There's a down side to early volume. If there are problems with the software or the manual, then lots of copies could mean lots of irritated owners and lots of bad publicity.)

At *Micro C* we don't make a cent off newsstands (let me tell you about newsstands), but we're there. Those copies are our version of inexpensive software packages. They give us exposure, credibility, new subscribers, and new advertisers.

They're our way of reaching the marketplace.

Jim Skinner arrives at SOG fly-in.

Speeding Up AT Floats

Sandy uses Autocad to do our schematics. It's a good package, but larger drawings (like the XT schematic) get pretty slow. (Five minute redraw on an 8 MHz 80186 with an 8 MHz 8087.)

So we put together a 12 MHz 80286 AT clone with an 8 MHz 80287 math chip and a super fast hard drive. Boy, did that puppy scream (*four* minute redraw).

Now hold on a minute. What's happening? This system costs twice as much as the other one. It's got to be faster than this!

About a week later Scott Baker called to tell me about his little adaptor board.

"If you have an 8 MHz 80287, this'll make it run 8 MHz."

Big deal, I've got a 12 MHz AT. Sure the 287's running slower than the 286, but it's no doubt already running faster than the 8 MHz that it's rated. (Intel has had trouble making fast 287s, so ATs have traditionally run their coprocessors significantly slower.)

"I'll send you some benchmark programs. Run them, you'll

Continued from page 74

be surprised."

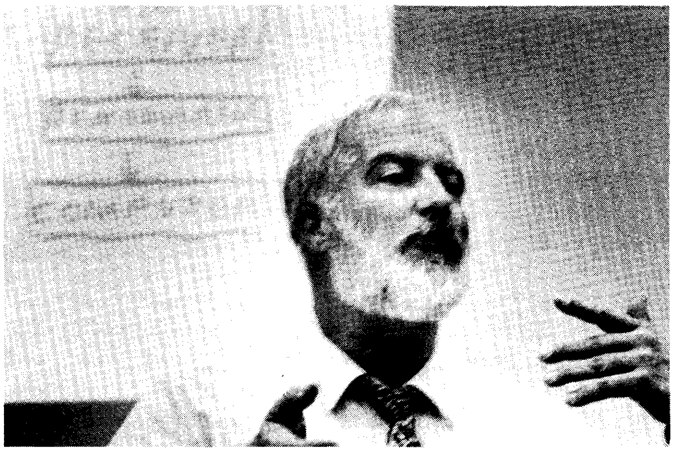
He did. I was.

My $250 80287 was loafing along at 6.2 MHz. At this rate it might as well have been a really cheap 6 MHz 80287. No wonder Sandy's plots were slow. Okay, I popped out the laggardly 80287, plugged in Scott's tiny adaptor board (it generates a 24 MHz clock which the 80287 divides by three), and plugged in the math chip. *Wow!* Only 3 1/2 minutes.

Oh well. At least that 287 is finally pulling its weight. I'm wondering how fast the 287s run in 6, 8, and 10 MHz clones. Those spendy chips are probably taking lunch breaks in the middle of transcendentals.

You'll find the benchmark programs in the issue #44 ARC file on the RBBS so you can see how your system is doing. You can get the 80287 speed up board for $29.95 from:

Sierra Circuit Design
18185 West Union Rd.
Portland, OR 97229
(503) 645-0734


Jim Warren

Exciting New IBM

IBM is one of a very few computer companies that isn't knee deep in back orders. In fact, IBM's been so successful at keeping up with demand, it's reportedly been able to shut down five assembly lines and lay off 900 employees.

However, that success may not last long. IBM has announced the model 35. It has a fancy new 80286 processor. It's AT compatible. And, I'm guessing that it'll have AT-compatible card slots. (IBM is strongly denying that it's backing down on the micro channel.)

Maybe they've created this dramatic new product because their PS-2 systems aren't very PC compatible. Maybe it's because micro channel cards are expensive. Maybe it's because the micro channel performance issues are moot since IBM's machines aren't particularly fast. But, it's probably because no one's writing software that runs exclusively on the PS-2. (I'm sure they were hoping.)

I took a lot of flack from some IBM employees for not being properly impressed with the PS-2 series. Now, I suppose they're ex-IBM employees. I'm sure they'll let me know.

Final Comments

An important part of SOG is the feedback we get from attendees. But attendees aren't necessarily representative of the rest of you. So, it would be great if you'd take a minute and fill out the survey card we've bound into this issue. There's a stack of three cards (probably near page 64). The bottom one is the survey.

In conjunction with the survey, I'm also interested in your life story. (With respect to computers, anyway.) Check out the short article in this issue on "Bits From Your Past."

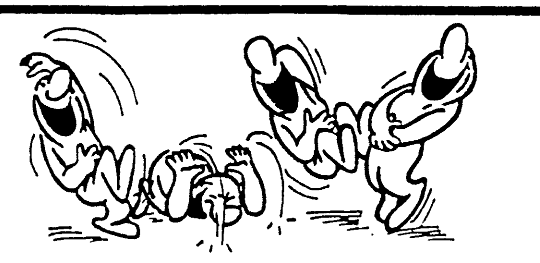
Also, be sure to see the SOG article by Barbara Hall. She wrote the piece for *Oregon Computer News*. It was so much better than anything we could do, we asked her if we could reprint it. She said "yes."

Finally, don't read Laine's column right after this one. I don't want to change our cover from "The Micro Technical Journal" to "The Travelog For International Computer Freaks."

David Thompson
Flighty Editor

[illegible]

♦ ♦ ♦



YOU WANT THE SOURCE?!

WELL NOW YOU CAN HAVE IT! The MASTERFUL DISASSEMBLER (MD86) will create MASM compatible source code from program files (EXE or COM). And the files are labeled and commented so they become USEABLE. MD86 is an interactive disassembler with an easy to use, word processor like interface (this is crucial for the REAL programs you want to disassemble). With its built-in help screens you won't have to constantly refer to the manual either (although there are valuable discussions on the ins and outs of disassembling which you won't want to miss).

MD86 is a professionally supported product and yet costs no more than "shareware". And of course, it's not copy protected. **VERSION 2 NOW AVAILABLE!**

MD86 V2 is ONLY $67.50 ($1.50 s&h) + tax

C.C. Software, 1907 Alvarado Ave., Walnut Creek, CA 94596, (415) 939-8153

Reader Service Number119

UNITS & MODULES

The Fun Has Just Begun

Michael S. Hunt
845 E. Wyeth
Pocatello, ID 83201
(208) 233-7539

With John Jones leaving the fold, I had a chance to expand the Pascal Column. (Not that John didn't stretch things a bit with his looks at Modula and his $6 scanner.) Fortunately, at SOG, Mike volunteered to do an algorithm column and this is it. (By the way, I understand the Army is famous for this kind of volunteering.)

Mike has a B.S. in Mathematics/Computer Science and he's currently a graduate student in mathematics and a teaching assistant at Idaho State University.

This column started at SOG. Dave was looking for a topic for a new column, and he mentioned it at the Micro C forum. I suggested an algorithms column. I didn't realize it then, but I had walked into a trap. Dave said in an hypnotic voice, "Would you be interested in writing the column?" I don't remember much after that. After my head cleared, I found myself sitting in the dark staring at the computer screen. Somewhere over my shoulder there lurked a deadline.

So, what can I say about algorithms? An algorithm is a finite sequence of instructions, each (hopefully) clear in meaning, that can be executed with a finite amount of effort in a finite amount of time. Algorithms run naturally through the code we write. Almost every article in *Micro C* has an algorithm. Even On Your Own and Around The Bend give us an approach or methodology for solving a problem.

Units & Modules

And what do units and modules have to do with algorithms? Units and modules are abstract machines. The internal workings of the machines or units can be hidden from the user. The user is only concerned with what each function or procedure does, not how it's done.

A unit or module attempts to be a collection of functions, procedures, and data types that do something. A well-designed unit should contain a well-designed algorithm. That way we create libraries of reusable abstract tools.

This column will deal primarily with algorithms, their design and usage. The module or unit concept provides an excellent vehicle for their presentation.

Directions

Most of my computer experience is from academia, not the real world. Academics spend a great deal of time studying and improving algorithms. But an algorithm is nothing more than a mind game if you don't have a real problem.

A good algorithm will often take advantage of peculiarities in the problem in order to find a solution.

Though no two problems are alike, many problems have similar characteristics. Often you can manipulate a problem into a different form so that you can use familiar tools to solve the problem. Radical and unorthodox thinking can provide you with a view of the problem you might not have normally seen. With practice, it gets easier to see similarities between the current problem and previous ones.

Each issue I'll throw out a programming problem. In the following issue I'll present at least one solution. The solution will be an algorithm that I hope will be more elegant than brute force.

Left to run amuck and out of control, I'm planning to cover the following:

The Black Art of Sorting
Stacks & Queues
Linked Lists
Strings & Character Manipulation
Tree Structures
Arrays & Matrices
Graphs & Networks
Search Strategies
Set Operations
Design & Analysis of Algorithms
Operations Research
Numerical Analysis
Mathematical Modeling

I'll try not to rehash previous *Micro C* articles but, where appropriate, I'll include references to other books and magazines.

All the code that I write will be available as a Turbo Pascal 4.0 unit and a Modula-2 module. I will use Turbo Pascal 4.0 as the language for the column because it will be most familiar to most of you. I am only a beginner C programmer but I will attempt to translate as much of the code to C as I can. The Turbo unit, Modula-2 module, and C code (when I can manage) will be available on the Micro C RBBS and the issue diskette.

Speak To Me

If you have a problem in need of a "better" solution, or if there's a topic you would like to see covered, please write. Otherwise, I will blindly forge onward believing you find the topics I cover useful. If you have a different or "better" solution to a problem, send it in.

In addition to dealing with algorithms, this column is a forum for any hints, tricks or tips you send in. If you have a piece of code that works for you, share it with the rest of us.

This Time

Since this is the first article, I have no previous material to build from. But here's a program (see Figure 1) to exercise your mind. This little brain teaser was obtained from David Wall at IFRICS, July 1987. The question is what does this program do? This is a good test of your code reading skills.

The program uses procedure passing in the parameter list and recursion to throw you off track. Procedure passing is the second feature that attracted me to Modula-2. The first was the power it gave me over large programs (5000+ lines).

The ability to pass procedures is an extremely powerful tool. When used carefully you can write data handling routines that are not data-type specific. The logic of a particular algorithm can be contained in a procedure or unit, but all the routines that are specific to a data type are passed in the parameter list.

This works particularly well for the swap and comparison routines in a sort algorithm. You can build a very flexible sort package by creating a set of swap and comparison routines for each data type you use. Then code several sort algorithms and design them to import the swap and comparison routines through the parameter list. Finally you use some selection criteria in an IF or CASE statement to call the correct sort package.

Contrary to popular myth, Quicksort is not the best sort for all data sets. So, I'll be covering the black art of sorting.

An algorithm is nothing more than a mind game if you don't have a real problem. A good algorithm will often take advantage of peculiarities in a problem...to find a solution.

Figure 1 — WhatAmI

```
Program WhatAmI(input,output);

  procedure Who(var prev: integer;
                procedure PrevWhat);
  var n : integer;

    procedure What;
    var temp : integer;
    begin (* what *)
      if n > prev then
      begin (* if *)
        temp := n;
        n := prev;
        prev := temp;
        PrevWhat;
      end; (* if *)
    end; (* what *)

  begin (* who *)
    if not eoln(input) then
    begin (* if *)
      read(n);
      What;
      Who(n,What);
      write(' ',n:1);
    end (* if *);
  end (* who *);

  procedure WhereInput;
  var endWho : integer;

    procedure EndWhat;
    begin
    end; (* endwhat *)

  begin (* whereinput *)
    endWho := MAXINT;
    Who(endWho,EndWhat);
    writeln;
  end; (* whereinput *)

begin
  WhereInput;
end. (* program *)
***
```

I changed a few identifiers in the WhatAmI program so they would not give away the function of the program. Follow the code carefully and take notes if necessary. There is a very subtle hint in the above paragraph. Next time I'll reveal the true nature of this program.

Coming Up

Even with the 1 MB on my PC Tech X16B or the 64 KB in my Kaypro 4-83, I still run out memory. This often happens when trying to run mini problems on a micro. Much of the information we store and retrieve in computer applications is redundant. (Let me repeat that ...) A method for mapping all of the redundant information into the same storage space could reduce memory requirements significantly.

This space savings doesn't come without a price, however. The additional algorithms and data structures require computational time and memory space. The savings comes when the space saved exceeds the overhead for the data structures. The algorithm running time can often be offset by the reduced amount of data to be maintained.

We could also devise a virtual storage system and swap some of the information to and from disk, but disk storage can also be limited. I have yet to write that great virtual storage system. Instead I will try reducing the amount of data to be stored.

A good example of redundant data is a spelling dictionary. Many words have the same prefixes and/or the same roots. Mapping all the redundant sequences would create a tangled maze of paths. So, I'll tackle the problem with linked lists and trees.

The problem is: given a list of words, how do we store them on disk (or in memory) so we can use them as a dictionary?

That also leads to other questions like: How do we check words in a text file? How do we add words to or delete words from the dictionary ... ?

Think about it. I'll take a look at this problem and tell you what "WhatAmI" does in the next exciting episode of "Units & Modules."

♦ ♦ ♦

Continued from page 6

RAM Speed Debate

I read with interest Gary Entsminger's comments in the Tidbits column (*Micro C* issue #42) regarding RAM chip speeds. Gary must have an uncanny way of choosing chips that fall on the very good end of the spectrum. Dave must also be quite good at it judging from his editorial comments.

I'm not as lucky. My experience is that with 1 wait state systems you need 150 nsec at 8 MHz, 120 nsec at 10 MHz, and 100 nsec at 12 MHz. It appears that the number of wait states is at least as critical as the system speed.

For instance, I have four banks of 120 nsec 256K chips (NECs, mind you) that run at 10 MHz, 1 wait state but not at 8 MHz, 0 wait states (theoretically, they should). I've also had a couple of banks of 120 nsec, 64K chips that wouldn't even boot at 8 MHz, 0 wait states but would come up (albeit crash after a short run due to parity errors) at 12 MHz, 1 wait state.

Incidentally, those who are thinking of building a fast AT clone might be better off buying a cheap XT clone board with 256K of slow 150 nsec RAM, and adding an Intel Inboard 386/PC.

It comes with 1 Meg of 32-bit memory and costs less than what you would pay for the 80386 and the memory chips by themselves. It runs applications an average of three times faster than an AT, even with its 8-bit I/O setup. Tricks like hard disk caching and remapping ROM and EGA BIOS into 32-bit memory help. It's almost as fast as a real 386 board, but costs a lot less money.

Hector Santos
2616 Berkeley Ave.
Los Angeles, CA 90026

Editor's note: Gary was optimistic, but his optimism isn't always unrewarded. I was conservative because I like to know my machine will work. I suspect something in your system's RAS, CAS, buffering or RAM select logic is a bit slow. That would make the system more demanding. (I haven't heard anything bad about the Japanese RAM parts.)

C Vs. ASM Imbalance

I wish to hastily object to Tim Berens' parting request (*Micro C*, issue #42, p. 6): "Please print a more balanced view of issues such as these in the future."

In the first place, Eric Isaacson cannot be wrong. He's from Indiana. Indiana people invented the television (Philo T. Farnsworth), the hand-held calculator (Bowmar Brain MX-10), and the home video game (Magnavox Odyssey). Eric's A86 assembler is so fantastic that it makes higher level languages obsolete — just as Jack Purdum's Eco-C88 compiler is so outstanding that nobody would ever think of using assembly.

In the second place, how many people read *Micro C* because it gives them a balanced view? Peter Norton describes C as an "industrial strength" language — not safe for households with children. I like to think of *Micro C* as the "industrial strength" magazine. For a balanced view there's always *Mother Earth News*. *Micro C* readers prefer the touched and sullied output of unbalanced minds.

I sure would have liked to take an afternoon off and drive up to SOG. How many miles north of South Bend are you? Incidentally, it's a 90 mile round trip to buy *Micro C* on the newsstand, so I've been meaning to send for a subscription. But I forgot to include the check before I sealed the envelope. Maybe next month.

Steven Thomas
6010 Southeast Willow Rd.
Warren, IN 46792

Editor's note: We've sent a couple of bicycle delivery kids to the south part of Bend to give you a genuine Micro C envelope for your check. None have succeeded. I suspect you're either confused (do you live in Peoria maybe?) or your town was misnamed.

DBASIC Post Mortem

I thought you might like to know what happened to the great DBASIC experiment. In two words: it flopped.

DBASIC worked very well. Even the reviewer who least liked it rated it above all the other BASICs for speed and for being bug-free. So it was an artistic success but a marketing disaster. I sold 70 copies at $40 each during its first four months on the market, and there was no evidence of growth at the end of those four months. What happened? A lot of things.

Giving a lot of copies to user groups backfired. DBASIC became known as the *free* BASIC. I'm still getting letters asking for free copies. I was charging $40 for single copies but giving 10-copy sets free to user groups (to get the software out). Unfortunately, single users were claiming to be user groups.

The other free BASIC, the one that comes with the ST, was awful. But most ST types don't program very often, so that's the one they use. Free BASIC competing with *another* "free" BASIC is a very tough way to make a living.

When the ST first came out, a lot of techies were buying it — just the right market for DBASIC. While I was busy turning DBASIC into a commercial product, those sales stopped and a new buyer came along: the drooling rock-shooter. Rock-shooters don't write programs.

DBASIC had a command line interface, just like MS-DOS and the older CP/M and Apple II machines. ST folk, like the Macintosh folk, are dedicated mouse/icon freaks and refuse to use a command line interface.

The biggest problem is that the ST, in the U.S., is a toy computer. DBASIC was intended to be a *useful* BASIC. That's why it had double-precision floating point and transcendentals.

There were lots of reasons why DBASIC failed; all of them were my fault; I did not correctly identify a market before blindly charging ahead. A market is a group of folks who are able and willing to buy a product or service. For instance, I can't stand the typical ST magazines. And yet, the magazines which support a computer tell you a lot about those who own it.

Early in December I filed for dissolution of DTACK Grounded Inc. and packed my bags. I left New Mexico at the end of the year and am now in Silicon Valley. DTACK did *not* go bankrupt; all the bills were paid. But it was obvious that the company could not survive, even with just one employee. I still get about one check a week for DBASIC. I send the checks back. I don't have a business license or manuals — most of them are in Santa Fe's municipal land fill.

Hal W. Hardenbergh
1111 W El Camino Real Ste 109-406
Sunnyvale, CA 94087

Editor's note: Hal, I'm curious about the rock-shooter. (Don't send one, we have enough extra stuff around the office.)

Beware Vaccine

While at the West Coast Computer Faire, I picked up a copy of Vaccine to protect my system from viruses.

The program examines the files on your hard disk — all executables and important data files. It keeps a catalog of the files along with CRC information. Then, before it allows a program to load, it verifies that the program hasn't been changed.

When I installed Vaccine on my AT, I ran into a few interesting problems. It seems that Vaccine uses the DOS COPY and FORMAT programs to make a master disk which you can use to authorize new programs. Well, when I inserted 360K disks into the HD drive, FORMAT and COPY failed but the Vaccine installation program continued to run without taking notice of the errors.

The result was that all programs on all disk drives were locked out. (I mean *all* programs.) The system was dead. I had a prompt but I couldn't run anything. The step of copying the master authorization file to the hard disk from the temporary floppy failed. The result was that no program was authorized to run.

I was in a panic. I thought, "The ultimate virus got me." Actually, the cure was to reboot from floppy and erase the Vaccine TSR from the hard drive's AUTOEXEC.BAT. When I went back and reread the manual, all it said was the floppies had to be "compatible" with the drive type. What they meant was a high density disk had to be used with HD drives.

Correcting this step, I reinstalled and this time it went smoothly. All COM and EXE files were checked and authorized in addition to the master data tables. As a further safeguard these files were set to read-only, making it that much harder for a virus to change things. When loading a file there was a slight delay while "... program checks OK ..." flashed on the screen but, other than that, everything worked just fine.

The main problem is that I do a lot of program development, and before I can run a newer version of a program, I must "authorize" it. This is a pain. After a few hours I de-installed Vaccine. For me, maybe the system worked too well.

This isn't meant to be a review, only some comments on what is actually a fairly good program (especially good for bulletin board SYSOPs, I imagine).

Clark A. Calkins
C.C. Software
1907 Alvarado Ave.
Walnut Creek, CA 94596

Editor's note: Tadas Osmolskis held a SYSOP session at SOG. The primary topic was viruses and Trojans, and I understand no one from the group had yet seen such a beasty. (We haven't.)

There appears to be some paranoia among the general population, however. We just had a customer return a group of Micro C disks because he found some remnants of erased files and some sector errors. He was absolutely certain we were trying to infect his system, especially when he noticed that files had been changed. (Hey, we've come a long way from the early days of public domain, but this is crazy.) As for the sector errors, we've always been ready to replace or refund.

A Pat On The Back

I've been reading your magazine off and on for a couple of years and have finally been moved by David Thompson's eloquent plea in *Micro C* #43. If I had known before that subscribing would encourage irresponsibility, funny articles and all that lot, I would have signed up a long time ago. I'm not really an anarchist, but that's just because I'm not organized enough.

I'd like to compliment all of your editors and contributors. *Micro C* is one of the few magazines that I ever read from cover to cover, every word, including the ads. In particular, Bruce Eckel's "Delving Into The Black Arts" (I built the board in slightly more than an hour, but the time passed quickly.), David Thompson's "Keeping Your Hard Drives Running," and all of Larry Fogg's PC support chip articles were absolutely first rate. Of course, everything else is too, but these articles stick in my memory, probably because I read each of them several times (refresh cycles).

In reference to the new format — looks good. The content doesn't seem to have suffered for the change. As for the advertisements and Ron Schroeder's comments in #43, things could be a lot worse. I offer as evidence Borland's recent "TurboMan" series in the front of *BYTE*. The ads in *Micro C* are not what I call the flashy 5th Avenue type, but rather tend towards a subdued black on white.

Keep up the good work.

Nathan Engle
6465 Piping Rock Lane #30
Indianapolis, IN 46254

Editor's note: Thanks for the kind words, Nathan. And, just to show how responsive we are to reader input, check out page 13 for our first ever, inside, four color ad!

Epsilon Info?

Two years ago a friend bought a used Data General One laptop (an early IBM PC compatible). Some software came with it, including an EMACS type of editor called Epsilon. It's similar to Perfect Writer, but loads a file into RAM without using a swap file.

The help file is somewhat incomplete, and there is no information about the origin of Epsilon: I don't even know if it's a public domain or commercial program.

Have any *Micro C* devotees heard of Epsilon? Can anyone tell me more about it?

Deborah Rose
4444 Yale Station
New Haven, CT 06520

♦ ♦ ♦

CPM NOTES

Kaypro Clock Fix

I'm writing in response to Dr. Yates' letter in *Micro C* issue #41. I assume, Dr. Yates, that your Kaypro 10 motherboard is similar to my Kaypro 1 motherboard. Even if not, what follows may be of use. (Kaypro seems to have gone out of their way to make their model designations thoroughly confusing.)

I bought a Kaypro 1 in December of last year from a liquidator. The production tag indicates that the machine was produced in November 1986, and I suspect that Kaypro made a final run to clear out its parts inventory. What's important is that this is probably the final version of their universal board. It incorporates certain changes (upgrades?) which don't appear on the Micro C schematic and are not documented in any articles.

As you probably know, the Kaypro 1 is really a 4-84 without the parts for the clock, modem, and SASI. So I decided to install the missing parts and document any differences between my board and the Micro C schematic. Enclosed are my efforts to date. (See Figure 1.) I believe I've covered most, if not all, of the differences. I only wish someone could check my work since circuit tracing is very tedious.

I'm assuming that you have access to a Micro C schematic. If not, pin 23 (PWR ON) of the clock chip was originally tied high directly. On my board, pin 23 ties high through an unidentified resistor (R53). An unmarked capacitor (C87) goes from pin 23 to ground. This capacitor is obviously an electrolytic.

Based on a reading of the MM58167A data sheets, past experience, and superstition, I settled on a 100K resistor and a 1.0 µF dipped tantalum. Also, I'm using a small button type lithium cell.

For CR6 I'm using a 1N270 germanium diode in order to minimize voltage drop. (Never heard of a "1001.") And I substituted a 6.8 µF dipped tantalum capacitor for C54. The schematic says it should be 0.1 µF. But I like the idea of a moderate-sized tantalum to achieve some degree of orderly voltage decay while still providing good bypass qualities. Besides, my board calls for an electrolytic at that location.

I'm self-educated in electronics and a pro might argue the wisdom of some of my choices. But my clock has performed flawlessly from day one. Not one error through numerous power up/downs. If your board already includes these changes, or if the addition of these components doesn't solve your problem, you might also try pulling pin 3 (WR) high through a 10K Ohm resistor.

If you're interested in the rest of what I've doped out, I hope my drawings are sufficiently clear. I haven't had time to really study all of what I've drawn, but on the surface it appears that Kaypro may have had an 8 MHz upgrade in mind for this board. If so, I'll definitely try for it after I've addressed the bus loading problem of the 84s.

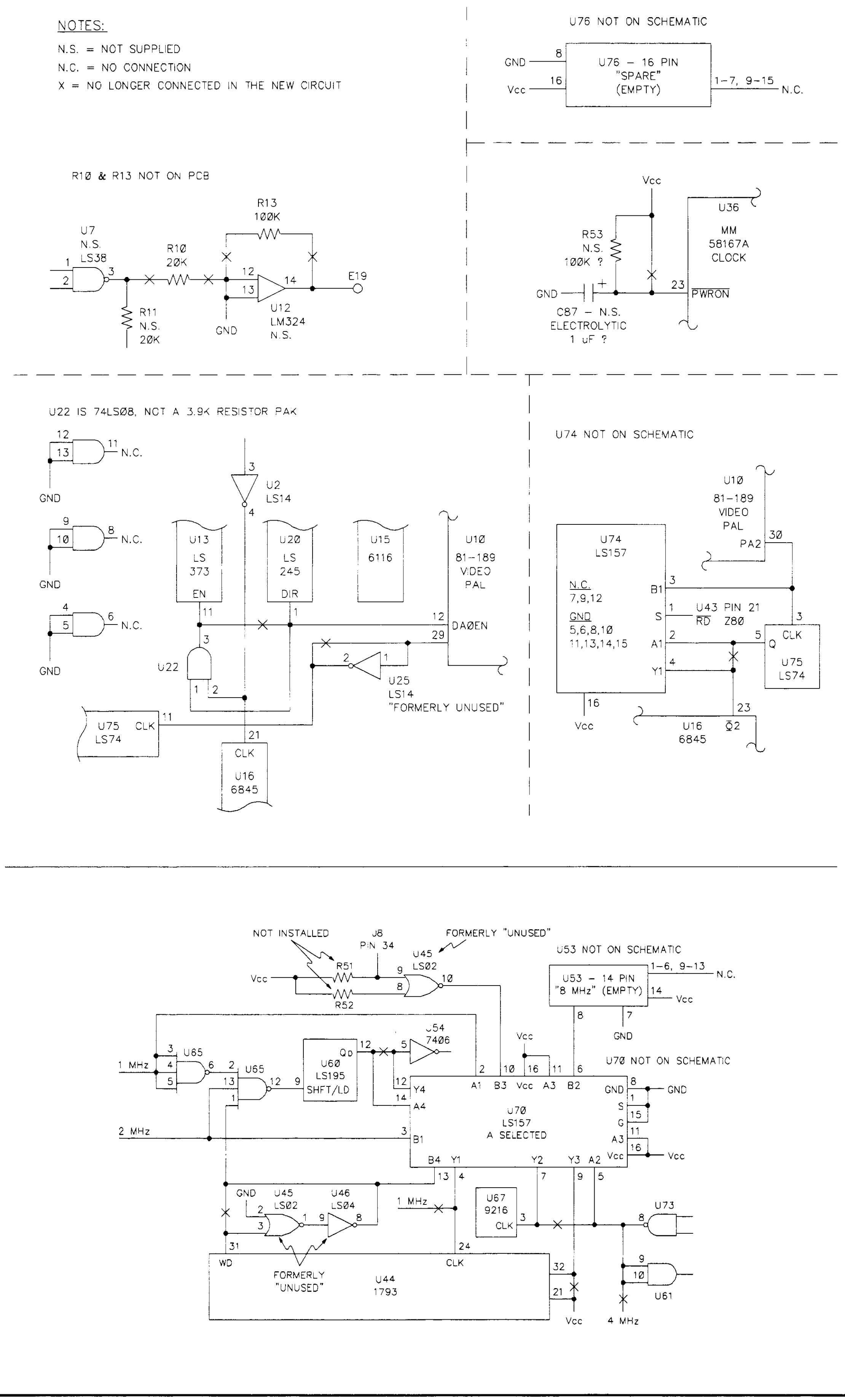
I'm self-educated in electronics and a pro might argue the wisdom of some of my choices.But my clock has performed flawlessly from day one.

Van S. Vangor
Bethlehem Tool
346C Retreat Rd.
Island Falls, ME 04747

Floundering Forth

Back in 1984 I got a copy of FIG Forth for the Kaypro II (from Micro Cornucopia) that had the extensions added by Bob Bumala and Kevin Appert. Recently I've been trying once

Figure 1 — Updates to MicroC Schematic for Universal Board (Final Version).



Unbelievable!

SOURCER™

- SEE HOW PROGRAMS WORK
- EASILY MODIFY PROGRAMS

SOURCER™ creates detailed commented source code and listings from memory and executable files. Built in data analyzer and simulator resolves data across multiple segments and provides detailed comments on interrupts and subfunctions, I/O ports and much more. Determines necessary assembler directives for reassembly. Complete support for 8088 through 80286, V20/V30, 8087, and 80287 instruction sets. We welcome comparisons with any other product, because no product comes close to the ease of use and output clarity of SOURCER.

On my list of programs that I simply won't do without!
—Robert Hummel, Senior Technical Editor, PC Magazine

SAMPLE OUTPUT

Fully automatic

Program header

Assembler directives

Determines data areas and type

Detailed comments

Simulator follows segment changes

Easy to read format

```
      resetprn.lst    ResetPRN v1.01               Sourcer Listing    19-Apr-88   4:05 pm      Page 1

                                PAGE  60,132
                                ;
                                ;                    RESETPRN
                                ;
                                ;        Created:   15-Apr-88
                                ;        Version:   1.01
                                ;

    = 0008                      data_le      equ     8                      ; (0040:0008=378h)

                                ;--------------------------------------------------------- seg_a -----
                                seg_a        segment para public
                                             assume cs:seg_a, ds:seg_a, ss:stack_seg_b

                                resetprn     proc    far
  658E:0000                     start:
  658E:0000  EB 23                           jmp     short loc_1
  658E:0002  52 65 73 65 74 50               db      'ResetPRN v1.01', 00h
  658E:0008  52 4E 20 76 31 2E
  658E:000E  30 31 0D
  658E:0011  0040               data_2       dw      40h
  658E:0013  0D 0A 52 65 73 65  data_3       db      0Dh, 0Ah, 'Reset Printer? $'
  658E:0019  74 20 50 72 69 6E
  658E:001F  74 65 72 3F 20 24
  658E:0025                     loc_1:
  658E:0025  0E                              push    cs
  658E:0026  1F                              pop     ds
  658E:0027 .BA 0013                         mov     dx,offset data_3  ; (658E:0013=0Dh)
  658E:002A  B4 09                           mov     ah,9
  658E:002C  CD 21                           int     21h               ; DOS Services  ah=function 09h
                                                                       ;  display char string at ds:dx
  658E:002E  B4 01                           mov     ah,1
  658E:0030  CD 21                           int     21h               ; DOS Services  ah=function 01h
                                                                       ;  get keybd char al, with echo
  658E:0032  3C 79                           cmp     al,79h            ; 'y'
  658E:0034  75 16                           jne     loc_3             ; Jump if not equal
  658E:0036  8E 1E 0011                      mov     ds,data_2         ; (658E:0011=40h)
  658E:003A  8B 16 0008                      mov     dx,ds:data_le     ; (0040:0008=378h)
  658E:003E  83 C2 02                        add     dx,2
  658E:0041  B0 08                           mov     al,8
  658E:0043  EE                              out     dx,al             ; port 37Ah, printer-2 control
                                                                       ;  al = 8, initialize printer
  658E:0044  B9 8000                         mov     cx,8000h
  658E:0047                     locloop_2:
  658E:0047  E2 FE                           loop    locloop_2         ; Loop if cx > 0
  658E:0049  B0 0C                           mov     al,0Ch
  658E:004B  EE                              out     dx,al             ; port 37Ah, printer-2 control
                                                                       ;  al = 0Ch, init & strobe off
  658E:004C                     loc_3:
  658E:004C  B4 4C                           mov     ah,4Ch            ; 'L'
  658E:004E  CD 21                           int     21h               ; DOS Services  ah=function 4Ch
                                                                       ;  terminate with al=return code
                                resetprn     endp
                                seg_a        ends

                                ;------------------------------------------------------ stack_seg_b ----
                                stack_seg_b  segment para stack
  6593:0000  00C0[FF]                        db      192 dup (0FFh)
                                stack_seg_b  ends

                                             end     start
```

(Source code output and inline cross reference can also be selected)

BIOS SOURCE

- CHANGE AND ADD FEATURES
- CLARIFY INTERFACES

for PS/2, AT, XT, PC, and Clones

The BIOS Pre-Processor™ with SOURCER provides the first means to obtain accurate legal source listings for any BIOS! Identifies entry points with full explanations. Resolves PS/2's multiple jumps for improved clarity. Provides highly descriptive labels such as "video_mode" and much more. Fully automatic.

SOURCER $99.95   BIOS Pre-Processor* $49.95   SOURCER w/BIOS Pre-Processor $139.95

USA Shipping & Handling $3; Outside USA $15; CA Residents add local sales tax 6, 6.5 or 7%; *requires SOURCER

All our products come with a 30 day money back satisfaction guarantee. Not copy protected. To order or receive additional information just call!

1-800-662-VCOM
(1-800-662-8266)

V COMMUNICATIONS
3031 Tisch Way, Suite 905, Dept. M3, San Jose, CA 95128 (408) 296-4224

PS 2, AT, XT, and PC are trademarks of IBM Corp.

Reader Service Number 62

more to get a handle on the language by working through the code in Steve Burnap's book on Forth. All was going well until I tried to use the PAD and TYPE words in the development of the records database. Trouble.

In plain FIG Forth, the statement "PAD 20 EXPECT <CR> Sgt. Pepper <CR>" causes PAD to accept characters until the <CR>. When "PAD 20 TYPE" is used to display the string, only the characters up to <CR> are displayed.

However, in the KForth derived from the extensions, the "PAD 20 TYPE" only displays "EFORTH_____________." Further examination of memory in the area reveals the same string no matter what is written to PAD by means of the EXPECT word. The string actually extends from "HERE 63 +" to "HERE 99 +," and consistently overwrites that part of PAD. The string is a group of five numbers followed by the word EFORTH and a series of dashes.

So I did some sleuthing.

After loading SCN 2 (which has some screen clearing words, a new LOAD, etc.), PAD works okay. But TYPE caused the screen to clear and the cursor to home!

Loading the editor (SCN 61-97) caused TYPE to display characters after the <CR> in the string (even if they were garbage), up to the number requested. SCNs 12-17 showed no change.

After SCN 18, the ERR/OOPS words, characters after the <CR> were represented by an underline. However, after the Magic Incantation (SCN 7) and the PATCHes to QUIT, FORGET, and ERROR, the command "PAD 20 TYPE" returned the aberrant string. In one attempt, the string didn't appear until I saved the new Forth after BYE.

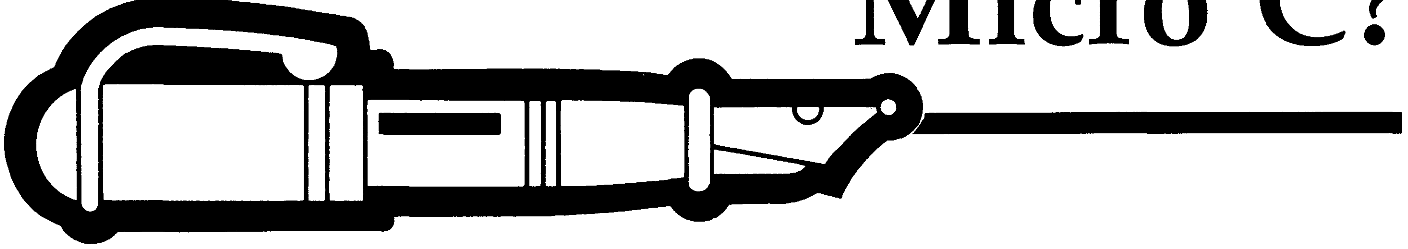
I'm not familiar enough with Forth to know why the PAD memory and TYPE word are corrupted by the additions (which, otherwise, I find most useful). I hope your resident Forth guru might shed some light on this problem.

Walter J. Rottenkolber
P.O. Box 936
Visalia, CA 93279

Editor's note: Sorry, our resident Forth guru wandered off years ago to live in a Wisconsin chicken coop. (No, I don't understand Forthers either.) But, there must be someone out there (cooped up, no doubt) who can send us an answer Forthwith.

♦ ♦ ♦

Want to write for Micro C?



Short Guide For Micro C Authors:

1. Read *Micro C*:
You'd be surprised how many people think we'd love to print their reviews of Apple II games, or their handy cheat sheet for WordStar 4.0. Fortunately, we're a technical journal so we're looking for people who are poking about along the front lines. (If you're doing something that hasn't been done before, or something very technical and very intense, holler.)

2. Communicate:
Articles that show up as a surprise generally don't get run (and if they do run, it's with significant revision). When David or Gary works with you, he can help you tailor the piece for *Micro C* or can point you to another, more appropriate publication. A majority of these 'unsurprising' articles get printed.

3. Be active:
The verb "to be" is overused and underpublished. Anytime you see "is" or "are" or "isn't" or "aren't," you're seeing passive verbs. Example: The parser is able to scan characters as they are input. Replace with: The parser scans keyboard characters.

4. Keep it meaty:
People read *Micro C* for information.

5. Keep it light:
(I know, I said "Keep it meaty.") You're writing from experience so let readers in on the frustration and the fun. We call ourselves "The Micro Technical Journal," but just between you and me we're really just technical people talking to our peers about exciting discoveries.

6. Use the first and second person.
First person is "I," such as the first line of #5 above. Second person is "you." "You" can be implied. "Keep it meaty!" implies the second person because I'm really saying, "You keep it meaty!" (And don't forget it!)

7. Short and Sweet:
Avoid long, twisted, windy, gargantuan sentences that run on and on, tied together by commas, ands, and buts, with lots of ideas — similar to, but not limited to, sentences such as this. (In other words, keep it simple.)
Keep your paragraphs simple too. A paragraph should contain a single, simple idea. We like short paragraphs, like this one.

8. Use subheads:
Subheads are very, very handy. They notify the reader of major shifts in subject, they're great for later reference, and they aid skimming.
Subheads also help you organize your writing. For instance, if you need to repeat the same subhead, or you can't come up with a single subhead that summarizes the material which follows, then you need to look at your organization.
I like to have a subhead every 15 to 30 lines (on the screen). That's not hard and fast: sometimes it's 5 lines, other times it's 40, but it averages out about every 20 lines.

9. Details:
Check your facts. When you're not absolutely sure about something, go to the source. We publish a lot of science fiction, at least a lot more than we'd like. Double check source listings. Does the copy you're including really compile? Or did you do some last minute cleaning up without testing? Did you include addresses of companies, phone numbers, prices, number of pages (if it's a book), etc? All those little details are important to the reader. Are they correct?

10. Finally, don't panic.
You're not writing a book. (Usually 9K to 20K bytes of text is fine.) And, it doesn't have to be perfect ... (Editors like to feel useful.)

Formats:

Articles should be submitted on disk or via Micro C's bulletin board (503) 382-7643, 300/1200/2400, 24 hrs.). We can read most disk formats except Apple, Rainbow, NorthStar, and Eagle CP/M, but please mark the format on the disk.

Your article will be easier to edit if you follow these guidelines:
1. Use standard ASCII format.
2. Use a single space between sentences.
3. Leave a blank line between paragraphs.
4. Don't indent paragraphs.
5. Don't right-justify your article.
6. Don't use hyphens except when they're part of a word or phrase. If you let your word processor add hyphens automatically, we have to remove them individually.
7. Use spaces, not tabs, even in listings .

KAYPRO COLUMN

Perfect Filer Utility

Extracting Data The Easy Way

By Joseph I. Mortensen
4214 Chelsea Ct.
Midland, MI 48640

If nothing else, the Perfect software series was controversial. Those who liked it swore by it. Everyone else avoided it like the plague. For all its idiosyncrasies, at least it was free with the Kaypro and people like Joe were soon writing patches and utilities to cover some shortcomings. Here's one.

Extracting data from Perfect Filer (the database which came with early Kaypros) is a painful, tedious process which nobody should have to do even once. (See my article in the July, 1988, issue of *PROFILES.*) It was an attempt to get Perfect Filer to generate a comma-delimited file for use with WordStar 4.0's MergePrint feature that finally forced me to act.

When I examined my Perfect Filer database using SuperZap, I found the information in Figure 1.

Figure 1 — Perfect Filer, Sector 0 Hex Dump

```
Address   00 01 02 03  04 05 06 07  08 09 0A 0B  0C 0D 0E 0F
------------------------------------------------------------
000000    7F 00 02 0D  58 00 00 00  00 00 00 00  00 00 00 00
000010    00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
 ......

/*** /
```

Sector zero serves as a file header. It's filled with 00H except for the first few bytes. Bytes 00 and 01 show the number of sectors in the file in the usual low byte, high byte order. This example has 007FH, so the file has 127 sectors of 128 bytes each. Bytes 04 and 05 contain the number of sectors in use (including sector 0), in this case 0058H or 88.

Sectors 1 and onward store the data. In unused (erased) sectors byte 00 is 0FFH. In active records byte 00 indicates how many sectors the record uses. Since Perfect Filer allows up to 1024 bytes (8 x 128 bytes/sector) per record, byte 00 can range from 01 to 08.

The following example has 01 at byte 00. Each field in a record ends with a null (00H). An empty field is indicated by a null as at 0D0CH in the example. 0FFH marks the end of a record unless the record fills an entire sector, in which case it ends with null. Any unused portion of a sector is filled with 0FFH. See Figure 2.

Having discovered how Perfect Filer stores data, I could proceed to write a program to convert each record and its several fields to the comma-delimited format used for data files by WordStar, MBASIC, dBASEII and many other programs. After conversion, the sample Perfect Filer record in Figure 2 would look like this (broken into more than one line to fit magazine margins): "Marguerite","","Bremer", "Mrs.", "Marg","","","636 Helicon, #285","","Minor Function","CA","92506","","619","5558341","x","","","","" ,"1","1","84".

Every field is enclosed with double quotes, fields are separated by commas, empty fields are represented by "", and the record ends with a carriage return. The finished data file consists of all the records from the Perfect Filer DATABASE.

An Easier Way

I finally found an easy way to extract data from Perfect Filer. It's a utility program called PF2ASCII.COM. (You can find it on the Micro C RBBS or the Issue #44 disk for $6 from Micro C.) It gets the names of input and output files either from the command line or from prompts after the program is running. It opens the Perfect Filer DATABASE and creates the output file. It then reads sector 0 to determine how many sectors the input file contains. Next it advances to the first data sector (at offset 80H) and reads byte 00.

If byte 00 is not 0FFH (marking an erased record), the program then writes the first double quotation mark to enclose the first field, reads and writes each printable ASCII character until it comes to a null (the end of field marker). It then inserts quote, comma, quote (",") to separate the fields and reads the next byte. If an empty field (indicated by a null byte) is encountered, the (",") are inserted in the output.

If the program finds the end of record marker (0FFH), it puts in the closing quote ("), adds a carriage return, and then moves to the next sector and starts the process all over. When all the sectors have been read and processed, PF2ASCII reports the job finished and closes both input and output files.

See Figure 3 for the main WHILE loop in PF2ASCII.COM (it's written in Turbo Modula-2).

Using PF2ASCII

PF2ASCII works with any Perfect Filer database and has distinct advantages over Perfect Filer's internal method of extraction (using the Generate List/Report option). With the latter the output must be formatted for each database. The number of fields you can extract is limited to 19 even though a Perfect Filer record may have up to 70 fields. PF2ASCII isn't limited.

The input file is always named DATABASE and is on the Perfect Filer data disk. If you do not enter input and output names on the command line then PF2ASCII prompts you for them.

The output works with WordStar 4.0's merge print which saves you the misery of preparing data files. Because the output is in ASCII form, data can be exported to MS-DOS programs which accept the comma-delimited format.

Why Modula-2?

Although I used Turbo Modula-2 to write PF2ASCII, such a short program could as easily have been coded in BASIC, C, or Pascal. I happened to have Modula-2 on hand and wanted more experience programming with it.

♦ ♦ ♦

Figure 2 — Perfect Filer Record

```
Address 00 01 02 03  04 05 06 07  08 09 0A 0B  0C 0D 0E 0F
-------------------------------------------------------------
000D00  01 4D 61 72  67 75 65 72  69 74 65 00  00 42 72 65 |.Marguerite..Bre|
000D10  76 65 72 00  4D 72 73 2E  00 4D 61 72  67 00 00 00 |mer.Mrs..Marg...|
000D20  36 33 36 20  48 65 6C 69  63 6F 6E 2C  20 23 32 38 |636 Helicon, #28|
000D30  35 00 00 4D  69 6E 6F 72  20 46 75 6E  63 74 69 6F |5..Minor Functio|
000D40  6E 00 43 41  00 39 32 35  30 36 00 00  36 31 39 00 |n.CA.92506..619.|
000D50  32 34 31 38  33 34 31 00  78 00 00 00  00 00 31 00 |5558341.x.....1.|
000D60  31 00 38 34  00 FF FF FF  FF FF FF FF  FF FF FF FF |1.84............|
000D70  FF FF FF FF  FF FF FF FF  FF FF FF FF  FF FF FF FF |................|

/***/
```

Figure 3 — PF2ASCII.COM Main WHILE Loop

```
WHILE SecNum<LONG(NumSecs) DO (* Keep going through whole file *)
   GotoXY(0,10);
   ClearToEOL;
   SecNum := SecNum + 1L;    (* Increment sector number *)
   SetPos(f1,SecNum*128L);   (* Go to beginning of each sector *)
                             (* Report progress *)
   WRITE('Reading......Converting Sector Number ',(SecNum));
   CharCount := 0;
   WHILE CharCount<128 DO    (* Read each byte in sector *)
      IF NOT EOF(f1) THEN ReadByte(f1,Ch);
        CASE Ch OF    (* If 0FFH is first byte in sector, skip to next
                       sector, otherwise put CR at end of record, bump
                       record counter and go to next sector. *)
          EOR : IF CharCount > 0 THEN WRITELN(f2); INC(NumRecs) END;
               CharCount := 128; |
    (* If character is NULL then check next charac-
                 ter.  Then back up file pointer one byte to
                 make sure no characters are skipped.
                 If character is 01 to 08 (indicating start of
                 new record) or 0FFH (indicating end of record),
                 add final comma and CR and go to next sector.
                 If it's a printable character, put "," to
                 separate fields.  Increment character count.
                 If NULL is last character in file, add final
                 double quote and CR, bump record counter, and
                 end processing. *)
         NULL : IF NOT EOF(f1) THEN ReadByte(f1,Ch);
                   SetPos(f1,NextPos(f1)-1L);
                   CASE Ch OF
                  01C..10C,EOR : WRITELN(f2,'"'); CharCount := 128;
                                 INC(NumRecs); |
                NULL, ' '..'~' : WRITE(f2,'","'); INC(CharCount);|
                  ELSE END; (* CASE *)
                ELSE IF EOF(f1) THEN
                  WRITELN(f2,'"'); CharCount := 128; INC(NumRecs)  END
                END; | (* IF NOT *)
                (* If character is printable, write it to output file
                   and increment character count. *)
      ' '..'~' : WRITE(f2,Ch);  INC(CharCount); |
                (* If character is 01 to 08 (indicating a new record)
                   write a double quote to file for first field. *)
      01C..10C : IF CharCount = 0 THEN  WRITE(f2,'"'); END;
              INC(CharCount); |
        (* If character is anything else, ignore it, and bump
           character count. *)
       ELSE INC(CharCount) END (* CASE *)
   END (* IF NOT EOF *)
 END  (* WHILE CharCount *)
END; (* WHILE SecNum *)

/***/
```

KAYPRO COLUMN

Perfect Filer Utility

Extracting Data The Easy Way

By Joseph I. Mortensen
4214 Chelsea Ct.
Midland, MI 48640

If nothing else, the Perfect software series was controversial. Those who liked it swore by it. Everyone else avoided it like the plague. For all its idiosyncrasies, at least it was free with the Kaypro and people like Joe were soon writing patches and utilities to cover some shortcomings. Here's one.

Extracting data from Perfect Filer (the database which came with early Kaypros) is a painful, tedious process which nobody should have to do even once. (See my article in the July, 1988, issue of *PROFILES.*) It was an attempt to get Perfect Filer to generate a comma-delimited file for use with WordStar 4.0's MergePrint feature that finally forced me to act.

When I examined my Perfect Filer database using SuperZap, I found the information in Figure 1.

Figure 1 — Perfect Filer, Sector 0 Hex Dump

```
Address  00 01 02 03  04 05 06 07  08 09 0A 0B  0C 0D 0E 0F
-----------------------------------------------------------
000000   7F 00 02 0D  58 00 00 00  00 00 00 00  00 00 00 00
000010   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
 ......

/*** /
```

Sector zero serves as a file header. It's filled with 00H except for the first few bytes. Bytes 00 and 01 show the number of sectors in the file in the usual low byte, high byte order. This example has 007FH, so the file has 127 sectors of 128 bytes each. Bytes 04 and 05 contain the number of sectors in use (including sector 0), in this case 0058H or 88.

Sectors 1 and onward store the data. In unused (erased) sectors byte 00 is 0FFH. In active records byte 00 indicates how many sectors the record uses. Since Perfect Filer allows up to 1024 bytes (8 x 128 bytes/sector) per record, byte 00 can range from 01 to 08.

The following example has 01 at byte 00. Each field in a record ends with a null (00H). An empty field is indicated by a null as at 0D0CH in the example. 0FFH marks the end of a record unless the record fills an entire sector, in which case it ends with null. Any unused portion of a sector is filled with 0FFH. See Figure 2.

Having discovered how Perfect Filer stores data, I could proceed to write a program to convert each record and its several fields to the comma-delimited format used for data files by WordStar, MBASIC, dBASEII and many other programs. After conversion, the sample Perfect Filer record in Figure 2 would look like this (broken into more than one line to fit magazine margins): "Marguerite","","Bremer", "Mrs.", "Marg","","","636 Helicon, #285","","Minor Function","CA","92506","","619","5558341","x","","","","" ,"1","1","84".

Every field is enclosed with double quotes, fields are separated by commas, empty fields are represented by "", and the record ends with a carriage return. The finished data file consists of all the records from the Perfect Filer DATABASE.

An Easier Way

I finally found an easy way to extract data from Perfect Filer. It's a utility program called PF2ASCII.COM. (You can find it on the Micro C RBBS or the Issue #44 disk for $6 from Micro C.) It gets the names of input and output files either from the command line or from prompts after the program is running. It opens the Perfect Filer DATABASE and creates the output file. It then reads sector 0 to determine how many sectors the input file contains. Next it advances to the first data sector (at offset 80H)

Figure 2 — Perfect Filer Record

```
Address 00 01 02 03  04 05 06 07  08 09 0A 0B  0C 0D 0E 0F
-----------------------------------------------------------
000D00  01 4D 61 72  67 75 65 72  69 74 65 00  00 42 72 65 |.Marguerite..Bre|
000D10  76 65 72 00  4D 72 73 2E  00 4D 61 72  67 00 00 00 |mer.Mrs..Marg...|
000D20  36 33 36 20  48 65 6C 69  63 6F 6E 2C  20 23 32 38 |636 Helicon, #28|
000D30  35 00 00 4D  69 6E 6F 72  20 46 75 6E  63 74 69 6F |5..Minor Functio|
000D40  6E 00 43 41  00 39 32 35  30 36 00 00  36 31 39 00 |n.CA.92506..619.|
000D50  32 34 31 38  33 34 31 00  78 00 00 00  00 00 31 00 |5558341.x.....1.|
000D60  31 00 38 34  00 FF FF FF  FF FF FF FF  FF FF FF FF |1.84............|
000D70  FF FF FF FF  FF FF FF FF  FF FF FF FF  FF FF FF FF |................|

/***/
```

Figure 3 — PF2ASCII.COM Main WHILE Loop

```
WHILE SecNum<LONG(NumSecs) DO (* Keep going through whole file *)
   GotoXY(0,10);
   ClearToEOL;
   SecNum := SecNum + 1L;    (* Increment sector number *)
   SetPos(f1,SecNum*128L);   (* Go to beginning of each sector *)
                             (* Report progress *)
   WRITE('Reading.......Converting Sector Number ',(SecNum));
   CharCount := 0;
   WHILE CharCount<128 DO    (* Read each byte in sector *)
      IF NOT EOF(f1) THEN ReadByte(f1,Ch);
        CASE Ch OF    (* If 0FFH is first byte in sector, skip to next
                         sector, otherwise put CR at end of record, bump
                         record counter and go to next sector. *)
          EOR : IF CharCount > 0 THEN WRITELN(f2); INC(NumRecs) END;
                CharCount := 128; |
   (* If character is NULL then check next charac-
                ter.  Then back up file pointer one byte to
                make sure no characters are skipped.
                If character is 01 to 08 (indicating start of
                new record) or 0FFH (indicating end of record),
                add final comma and CR and go to next sector.
                If it's a printable character, put "," to
                separate fields.  Increment character count.
                If NULL is last character in file, add final
                double quote and CR, bump record counter, and
                end processing. *)
         NULL : IF NOT EOF(f1) THEN ReadByte(f1,Ch);
                   SetPos(f1,NextPos(f1)-1L);
                   CASE Ch OF
                  01C..10C,EOR : WRITELN(f2,'"'); CharCount := 128;
                                 INC(NumRecs); |
                NULL, ' '..'~' : WRITE(f2,'","'); INC(CharCount);|
                  ELSE END; (* CASE *)
                ELSE IF EOF(f1) THEN
                  WRITELN(f2,'"'); CharCount := 128; INC(NumRecs)  END
                END; | (* IF NOT *)
                (* If character is printable, write it to output file
                   and increment character count. *)
      ' '..'~' : WRITE(f2,Ch);  INC(CharCount); |
                (* If character is 01 to 08 (indicating a new record)
                   write a double quote to file for first field. *)
      01C..10C : IF CharCount = 0 THEN  WRITE(f2,'"'); END;
            INC(CharCount); |
        (* If character is anything else, ignore it, and bump
           character count. *)
        ELSE INC(CharCount) END (* CASE *)
   END (* IF NOT EOF *)
  END  (* WHILE CharCount *)
END; (* WHILE SecNum *)

/***/
```

and reads byte 00.

If byte 00 is not 0FFH (marking an erased record), the program then writes the first double quotation mark to enclose the first field, reads and writes each printable ASCII character until it comes to a null (the end of field marker). It then inserts quote, comma, quote (",") to separate the fields and reads the next byte. If an empty field (indicated by a null byte) is encountered, the (",") are inserted in the output.

If the program finds the end of record marker (0FFH), it puts in the closing quote ("), adds a carriage return, and then moves to the next sector and starts the process all over. When all the sectors have been read and processed, PF2ASCII reports the job finished and closes both input and output files.

See Figure 3 for the main WHILE loop in PF2ASCII.COM (it's written in Turbo Modula-2).

Using PF2ASCII

PF2ASCII works with any Perfect Filer database and has distinct advantages over Perfect Filer's internal method of extraction (using the Generate List/Report option). With the latter the output must be formatted for each database. The number of fields you can extract is limited to 19 even though a Perfect Filer record may have up to 70 fields. PF2ASCII isn't limited.

The input file is always named DATABASE and is on the Perfect Filer data disk. If you do not enter input and output names on the command line then PF2ASCII prompts you for them.

The output works with WordStar 4.0's merge print which saves you the misery of preparing data files. Because the output is in ASCII form, data can be exported to MS-DOS programs which accept the comma-delimited format.

Why Modula-2?

Although I used Turbo Modula-2 to write PF2ASCII, such a short program could as easily have been coded in BASIC, C, or Pascal. I happened to have Modula-2 on hand and wanted more experience programming with it.

♦ ♦ ♦

0 1 1
1 0 0
1 0 0

TIDBITS

SOG VII, Instant Replay, Tiny Einstein, & Statgraphics

By Gary Entsminger
1912 Haussler Dr.
Davis, CA 95616

SOG VII highlights, a demo producer, and statistics. Gary pretends he's the Micro C reviewer. (I don't dare say too much lest I become a statistic myself.)

Well, it happened again. July came, and along with it 300 or so of you made the annual pilgrimage to Bend for our seventh annual semi-official micro-technical get-together. Thanks to those of you who made it another fun and energized event.

This year's SOG was perhaps our most relaxed ever (it looks like the Intel and Motorola fanatics are going to get along after all), with four days of blue sky and sunny weather.

We were especially pleased to have 11 new and stimulating speakers, as well as a dozen old favorites. Special thanks to —

Louis Baker	**Scott Ladd**
Andy Bakkers	**Paul Lamar**
Joe Bartel	**Greg Lobdell**
Ken Berry	**Tom Ochs**
Earl Brabandt	**Mike Sequiera**
Bill Davidson	**Reese Shepard**
Dan Doerr	**Willy Steiger**
Tom Domagala	**Paul Voda**
Bruce Eckel	**Michael Vore**
Chuck Forsberg	**Jim Warren**
Allyn Franklin	**Bill Weinman**
Earl Hinrichs	

Although I don't have room here to discuss all the talks, I'd like to at least mention three of my favorites —

Mike Sequiera's Chaos 101 was a particularly illuminating introduction to chaos theory. Mike, a mathematics instructor at Central Oregon Community College (in Bend) took a standing room only crowd of programmers and engineers into the depths where order and complexity give way to utter chaos.

Mike used movie cameras, videos, gadgets, slides, equations, photographs, and stimulating conversation to open up the world of fractals and dimensional bizarrity to even the most uninitiated. Mike's talk certainly stimulated some pretty strange conversations around greater Bend.

Paul Voda, author of the new programming language, Trilogy, was another sure fire hit. His discussion of programming languages in general and his (and others) attempts at creating a logical programming language showed us the logical limitations of all our favorite languages.

Paul Voda explaining his new logical language.

Trilogy, in my opinion, still has a ways to go before it's ideal, but even in its current state deserves serious attention from programmers needing a more logical environment for development. I'll be talking more about Trilogy in future issues of *Micro C*.

And Tom Ochs, creator of the equation solver, "Solve It," and president of Structured Scientific Software, really opened our eyes to the very subtle problems involved in developing numerical applications.

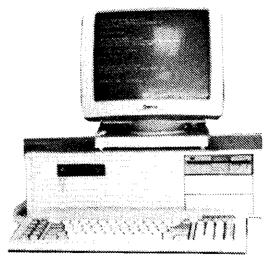
Tom, a contributor to *Micro C* and another mathematician, led us deep into the world of round-off errors, approximations, and "real" attempts to deal with chaotic behavior in solving engineering problems. Things, it seems, are not always as they seem. (A seemly comment.)

Special Christmas Dreams...

The Dream-286
40 MB AT Compatible

6-10 MHz 0 wait state motherboard
640 K of on board RAM
1 5.25 " TEAC 1.2M floppy drive
40 Mb Seagate ST-251 hard drive
Hard/floppy disk controller card
12" amber Samsung monitor (tilt/swivel)
Hercules compatible monographics card
AT style keyboard
AT case (UL and FCC approved)

Complete! **$ 1495**



The Dream-386

Quality 80386 based motherboard
1 Megabyte 80 ns on board RAM
2 Parallel ports and 2 serial ports
640 x 480 on board EGA/VGA card
1.2 Megabyte 5.25" floppy drive
Western Digital WA-2 hard/floppy controller card
Professional enhanced (101 key) keyboard
Case (UL and FCC approved; reset switch, power and turbo LEDs, keyboard lock

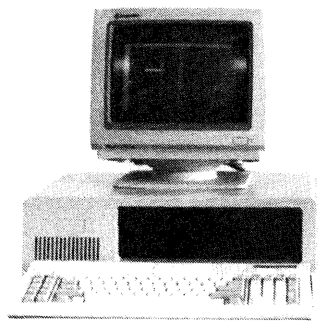
Complete! **$ 2295**

Novell Networks
IBM PS/2 Systems
Apple MacIntosh SE

The Dream-88
30 MB XT Compatible

XT 10 MHz Turbo motherboard
256 K of on board RAM
1 5.25 " 360K Fujitsu floppy drive
30 Mb Seagate ST-238 Hard drive
Floppy controller card (controls 2)
Hard disk controller card
12" amber Samsung monitor (tilt/swivel)
Hercules compatible mono graphics card
AT style keyboard
XT slide case (UL and FCC approved)

Complete! **$ 795**



Hard Drives

Seagate 20 MB w/controller ..	$ 269
Seagate 30 MB w/controller ..	$ 299
Seagate 40 MB w/controller ..	$ 345
Micropolis 71MB Hard Disk ..	$ 600
Seagate 80 MB ST-4096	$ 650

Floppy Drives

Fujitsu 360K 5.25"	$ 79
Fujitsu 1.2 MB 5.25"	$ 99
Sony 720K 3.50"	$115
Sony 1.44 MB 3.50"	$145

Floppy drives come with mounts

Citizen Printers

180-D (180 cps 9 pin)	$ 189
MSP-40 (260 cps 9 pin)	$ 339
MSP-45 (wide carriage)	$ 545
Tribute 124 (200 cps 24 pin)	$ 545
Tribute 224 (wide carriage) ...	$ 689
Premiere 35 (daisy wheel)	$ 499
Alps Allegro (24 pin)	$ 389

Tape Backup Units

Everex internal cassette	$ 595
Everex internal cartridge ...	$ 689

Both are 60 MByte units

Accessories

PC Mouse w/ paint	$119
C-7 Logitech mouse	$ 89
Logitech hirez bus mouse	$125
Kraft IBM/Apple Joystick	$ 25
Disk holder (100) w/keylock .	$ 15

Communications

1200 Baud internal	$ 75
2400 Baud internal	$145
1200 Baud external (w/ cable)	$ 99
2400 Baud external (w/ cable)	$235
PC FAX (card & software) ...	$389
Complete PC voice mail card .	$349

Software

Borland Turbo C	$ 79
Borland Turbo Pascal	$ 69
Borland Turbo Basic	$ 69
Borland Quattro	$189
Microsoft Works	$149
Microsoft Windows v2.03	$ 69
Microsoft DOS 3.21	$ 69
Microsoft DOS 3.30	$ 99
Microsoft DOS 4.0	$135
Norton Utilities (Advanced) .	$ 89
Norton Commander	$ 45

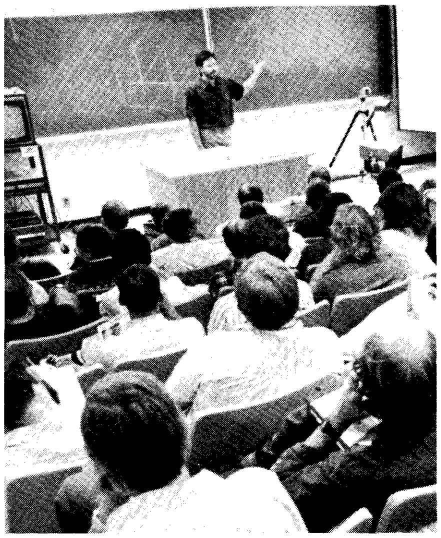
DreamTech
5175 Moorpark Avenue
San Jose, CA 95129
Open Monday - Saturday
10 AM to 7 PM
Phone: (408) 996 - 2373

MasterCard VISA

All systems carry a full 1 year warranty (original system configuration only). Prices are subject to change without notice. All orders are shipped UPS FOB San Jose unless otherwise specified at time of order. California residents add 7% sales tax. All registered trademarks are recognized. Dream-88, Dream-286, and Dream-386 are trademarks of DreamTech Computers.

Reader Service Number 16

If you're working on a numerical application, I suggest you get in touch with Tom: if he hasn't solved your problem, he's probably at least encountered and mulled over it.



Bruce Eckel discussing latest C++

Instant Replay & Tiny Einstein

As many of you know, I've been marketing an expert development system since early May.

One of the first problems I encountered, and am still wrestling with, is how to circulate information about Tiny to prospective buyers. The usual method (at least judging from the news releases and propaganda we receive at *Micro C*) is to mail a page or two of hype.

Although this is reasonably cost effective, it lacks the excitement of the program itself. You've probably noticed that we (*Micro C*) never (in contrast to most other micro computer publications) publish news releases. In fact, I rarely get even a twinge of excitement from a news release.

Enter Instant Replay, a relatively new programming idea from Nostradamus in Salt Lake City, that looks like it'll save me a bunch of time and has already begun to solve my excitement problem.

Instant Replay is a demo producer that's out-of-sight. You can use it three ways —

(1) to produce a prototype of a nonexistent program;

(2) to capture screens from either an existing program or to create screens for a potential program;

(3) to simulate the actual execution of an existing program.

The screen capture utilities let you grab screens and then edit them, inserting prompts for users, timed sequences, etc. This allows you to create some very effective demos and tutorials.

The simulation utilities are (effectively) keystroke recorders. To use them, you run your program and *use it*. Instant Replay records your keystrokes in a macro, which you then play back as a demo or tutorial. As with the screen capture utility, you can edit the keystroke macro, insert prompts, and change the timing of sequences.

This is an excellent, easy-to-use, reasonably-priced ($149.95) program with many possibilities. For more information about Instant Replay contact —

Nostradamus
3191 South Valley St. (Suite 252)
Salt Lake City, UT 84109
(801) 487-9662

For a demonstration/tutorial of Tiny Einstein created with Instant Replay contact —

Acquired Intelligence
P.O. Box 2091
Davis, CA 95617
(916) 753-4704

Relaxing at SOG.

For some time now, Alison (my wife) and I have been looking for a good statistics package. She's a graduate student at UC Davis, raised on mainframes, and tough to convince that a PC can manipulate the reams of data she accumulates each year.

Although it's still not clear that we've found the perfect replacement for the mainframe, we do think we've found an excellent alternative — Statgraphics from STSC.

Statgraphics is written in APL and will perform just about any statistical function you can think of —

- ANOVA
- nested ANOVA
- multiple regression
- non-linear regression
- principal component analysis
- Kolmogorov-Smirnov tests
- Kruskal-Wallis analysis by ranks
- Friedman two-way analysis by ranks
- time series analysis
- etc.

as well as many math functions —

- numerical differentiation
- fast Fourier transforms
- solutions of simultaneous equations
- prime number generation
- integer factorization
- etc.

Statgraphics is fast (we can compute ANOVAs for 1,000 or so datapoints in a few seconds on our Rabbit AT), menu-driven, and includes excellent graphics utilities to allow you to produce manuscript quality lines and pix.

Editor's note: A Rabbit AT is a hare faster than other ATs, except when doing turtle graphics.

You can plot histograms, barcharts, pies, scatterplots, and multiple x-y-z plots. You can scale graphs, include confidence limits, change the color and size of text, and even decide on which side of the axes you want your (major and minor) ticks!

In addition to having access to numerous statistical and graphical functions, you can also export and import dBASE, DIF, LOTUS, and TEXT files. In fact, although Statgraphics includes a reasonably good database manager, we choose to maintain our data with Paradox and Tiny Einstein and then export it to Statgraphics when we need to analyze something statistically.

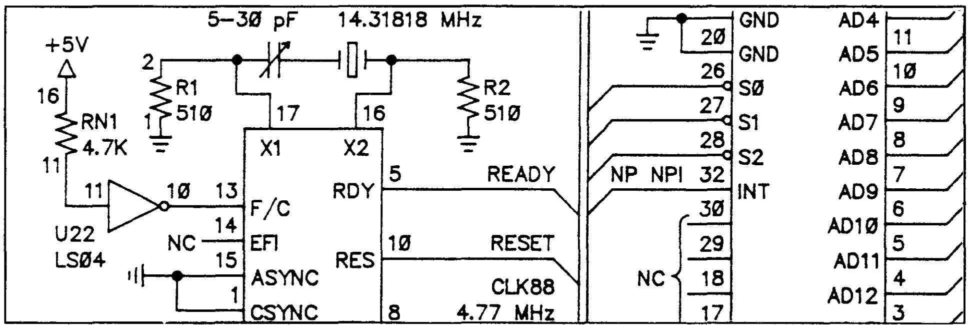
So far, this is the best statistical package we've found, and I highly recommend it for serious statistical use. At $895 it isn't cheap, but includes excellent support (the tech folks take your number and call you back) and seems fair-priced considering the quality and breadth of the program.

For more info —

STSC, Inc.
2115 East Jefferson St.
Rockville, MD 20852

♦ ♦ ♦

XT SCHEMATIC



At last you can plumb the mysteries of your computer with this single sheet schematic of the IBM XT's main board. A wealth of information for both True Blue *and* clone owners.

Need to know just how a non-maskable interrupt occurs (and how to mask it)? Is your keyboard dead (or do you just want to know how to disable it)? A trip through our schematic will answer your questions.

Although clones use slightly altered board layouts and different chip location names, they're close enough to the original for this schematic to be very useful. As an example — you have a dead clone. Lil sucker won't even beep. A look at the schematic shows the location of parallel port A. You know that the power on self test loads a checkpoint number into port A before each test. So now all you have to do is read port A with a logic probe to see how far the system went before it puked.

We'll include a list of these checkpoint numbers and some other pertinent trouble shooting information with the schematic.

IBM PC-XT Schematic **$15.00**

CP/M KAYPRO SCHEMATICS

Of course, we still provide a complete schematic of the processor board in your CP/M Kaypro. It's logically laid out on a single 24" by 36" sheet and comes complete with an illustrated theory of operation that's keyed to the schematic. You get detailed information available nowhere else.

For instance, those of you with the 10 and newer 84 systems get a thorough run down of the processor board's video section complete with sample driver routines. All packages contain serial and parallel port details and programming examples. Also coverage of the processor, clock, I/O, and disk controller (information that's not even available in Kaypro's own dealer service manual!).

Kaypro II & IV (pre-84) **$20.00**

Kaypro 10 (without modem) **$20.00**

Kaypro 2, 4, and 10 (84 series) **$20.00**

NOTE: These packages cover *only* the main boards. You're on your own when it comes to disk drives, power supplies, video cards, etc.

Phone Orders: (503) 382-5060 or 1-800-888-8087 Mon-Friday 9 am-5 pm PST

Mail Orders: P.O. Box 223, Bend, Oregon 97709

Techtips

Revised 386 Fractal Code

The way I coded the transform function for FR386 (see *Micro C*, issue #43, pg. 22) is inefficient, slightly incorrect, and opaque. I've written two new versions of the function; one is in 16-bit 80x86 code (see Figure 1), and the other is in 386 code (see Figure 2). The 386 version illustrates use of the "bsr" instruction.

transform() has very little effect on the speed of the fractal program, but I don't want the old version of the function to stand as an example of bad coding. I'm reminded of a familiar college experience — turning in an exam and realizing a split second later exactly what I did wrong. How embarrassing!

So far, FR386 has run on 16 and 20 MHz Compaqs (with both Herc and EGA/VGA graphics), and on 16 and 20 MHz PS/2 80s (standard VGA and IBM's super-duper Hi-res VGA). I did note one problem. If a computer has a Herc board but you haven't run the program to enable graphics, the is_herc() function returns a nonzero value and FR386 starts writing to the screen in character mode.

Harlan Stockman
5308 Noreen Dr. NE
Albuquerque, NM 87111

EGA Screen Save/Restore

If you want to save or restore a high resolu-

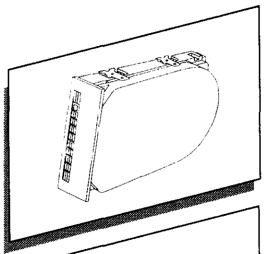
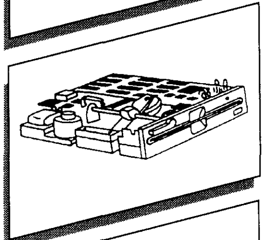
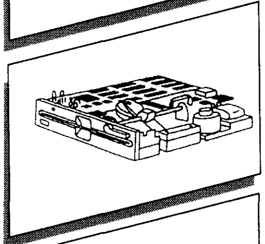
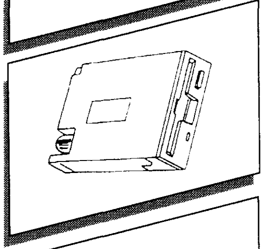
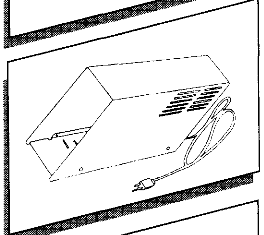
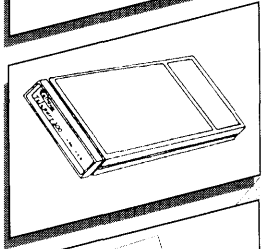
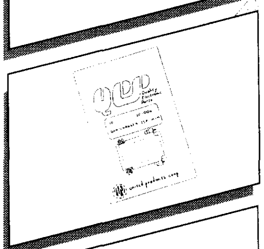
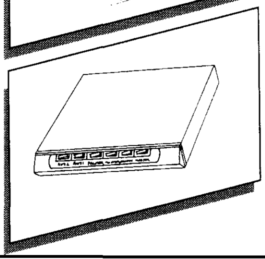
Figure 1 — 80x86 transform Function

```
   cseg
public  transform_
transform_:     pop di            ;return address
                pop cx            ;iter
                pop bx            ;switchpt
                sub sp,4          ;required by DeSmet for short return
                cmp cx,bx
                ja do_loop
                jmp di            ;short return
        do_loop:mov ax,bx         ;init color (cx) with switchpt
                sub cx,bx         ;cx now iter-switchpt
        loopit: inc ax
                shr cx,1
                jnz loopit
                jmp di            ;short return, color in ax
```

Figure 2 — 80386 transform Function

```
   cseg
public  transf386_
transf386_:     db 67h,8bh,44h,24h,02h ;mov ax,word ptr [esp+2] ;iter
                db 67h,8bh,4ch,24h,04h ;mov cx,word ptr [esp+4] ;switchpt
                cmp ax,cx
                ja do_bsr
                ret
        do_bsr: sub ax,cx
                db 0fh,0bdh,0c0h        ;bsr ax,ax
                add ax,cx
                inc ax
                ret
```

(continued on page 92)

20 MEG HARD DRIVES

SEAGATE Model ST-225
(Full 6 Mo. Warranty)*less face plate* $219.95

Western Digital Controller For Model ST-225 Drive
$69.95 w/cable

Yes!! These are for IBM compatible machines.

SHUGART # 465

QUAD DENSITY, 720 K, 96 TPI
1/2 HEIGHT
BLACK FACE PLATE
(NEW BUT HAVE COSMETIC BLEMISHES)
SOLD AS IS $49.95

TOSHIBA 5 1/4" FLOPPY DRIVES

DSDD, 360 K
1/2 HEIGHT
BLACK FACE PLATE
(90 DAY WARRANTY) $89.95

TOSHIBA 3.5" DISK DRIVES
IBM compatible
720K Byte
Double sided
Mounting kit
Power & Data cable adapter
Model # FDD 4210GOK

FULL 90 DAY WARRANTY $119.95

5.25" DISK DRIVE CABINETS
(with power supply)

For 1 full height or 2 half height
#CAB-25V5
$99.95 each

300 BAUD SMART MODEM (DIRECT CONNECT)
LOW COST SERIAL MODEM
7 x 10 x 1 1/2
$19.95

CABLES-CONNECTORS-SEX CHANGERS

"XT" STANDARD PARALLEL (10. DB-25M TO CN36M)	#10-004	$12.95
AC POWER CORD (6 COMPUTER STANDARD. MOLDED RUGS)	#10-001	2.95
RS232 MODEM ADAPTER (DB-25 M-F. PINS 2 & 3 REVERSED)	#20-001	9.95
SEX CHANGER, DB-25 F TO F	#20-005	9.95
SEX CHANGER, DB-25 M TO M	#20-006	9.95

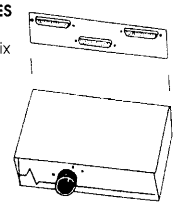
COMPUTER POWER SUPERVISOR (A LOW COST INSURANCE POLICY
PROVIDES SURGE/SPIKE PROTECTION
FOR YOUR COMPUTER!
ALL INCOMING POWER IS MONITORED & CONDITIONED BEFORE ALLOWING IT TO POSSIBLY TRASH YOUR COMPUTER OR OTHER SUCH VUNERABLE & EXPENSIVE EQUIPMENT.

- 5 OUTLETS, EACH WITH A LIGHTED SWITCH
- 1 MASTER ON/OFF SWITCH
- SYTLISH CABINET PROVIDES FULL SHEILDING
- RATED 15 AMP, 125 VAC, 1875 WATTS

DIM.12.5" x 2.5" x 14" Regular price $69.95 Your price $49.95

5.25" HARD DISK
CONTROLLER CARD
FOR ANY MODERN 5.25" HARD DRIVE

$89.95 W/CABLE



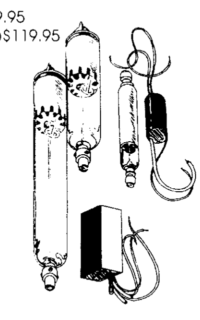
COMPUTER A-B SWITCH BOXES

#CN36 A-B Parallel Centronix
$24.95 each

#DB25 A-B Serial RS-232
$24.95 each

STEPPER MOTORS

Copal #SP-57
1/4" Shaft, 7.5 deg./step, 36 Ohm, 12VDC $6.95
3/16" Shaft, 35 oz." torque, 2.1A., 5VDC
WITH BRASS GEAR, 20 TEETH, 1/2" DIAMETER $9.95



LASERS

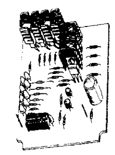
5 MW Laser Tube $89.95
Power Supply Kit (115VAC)$69.95
Power supply (wired) (12VDC)$119.95

1 MW Laser Tube$119.95
Power Supply (12VDC)$99.95

(These lasers are brand new and guaranteed to have a cosmetic defect or not meet manufactures full specifications. All are tested in our lab to insure your satisfaction.)

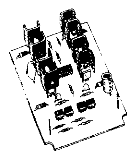
WARNING Voltages present and used by lasers can be lethal...Permanent eye damage could result from direct exposure to an on coming laser beam. Only those persons qualified to handle such potentials should do so.

TECHNA-KIT



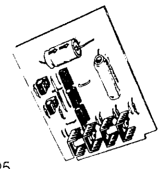
D-C Motor Controller
- Control 2 D-C motors with a computer or other logic source
- For motors rated 6-24 VDC
- Control forward/reverse/run/cw/ccw/stop
- Up to 6 Amp starting surge. 4 Amp cont.
- Dynamic breaking (capable)
- Will also run most 4-lead stepper motors

$29.95



USMD-C
- Control standard 6-lead stepper motors with a computer or other logic source
- For motors rated 1.7 - 12.0 VDC
- Optical isolation
- Control: forward/reverse/step rate/stop
- Industry standard 22 pin edge card connector

$29.95



COMPUTER AUTOMATE
- Use your computer to provide automation
- 8 separate driver ports per card
- 8 TTL/Cmos inputs
- 1 user defined sense switch
- 6-24 VDC
- 4 Amps/driver (max current)

$29.95

united products corporation
DISTRIBUTORS OF ELECTRONICS SINCE 1968
1123 VALLEY STREET • SEATTLE, WA 98109-4425

VISA AMERICAN EXPRESS

***PHONE:* (206) 682-5025**
***FAX:* (206) 682-5593**
M-F 9-6 SAT 9-5

Reader Service Number 101

tion EGA screen quickly (Mandelbrot sets, for example), there is a relatively simple way to do it. The computer can only "see" one bit plane at a time. So, to save or restore a screen, you need to modify the EGA's latching registers to access each of the bit planes.

The video controller can then map the four planes, one at a time, into the PC's 64K video window. Thus you can create a screen save/restore with a simple fwrite or fread to or from screen memory.

I set the latching registers for each plane with two outp statements, and because the resulting routines are short they're really fast. On an 8 MHz XT clone, the save_screen function takes five seconds for an entire screen. get_screen takes a mere four seconds to do the restore. See Figure 3 for both functions.

For the high-res EGA mode (640 X 350 with 16 colors), the file produced is 115,200 bytes. save_screen and get_screen can also be adapted to work with other EGA modes. To save a 640 X 200, 16 color EGA screen, the "28000" parameter of all fwrite and fread functions should be changed to "16000." This produces a 64K screen file. For a 320 X 200, 16 color screen, use "8000." This results in a 32K file.

These functions should compile under both QuickC and Turbo C. Use either the large or compact memory models because of the use of far pointers.

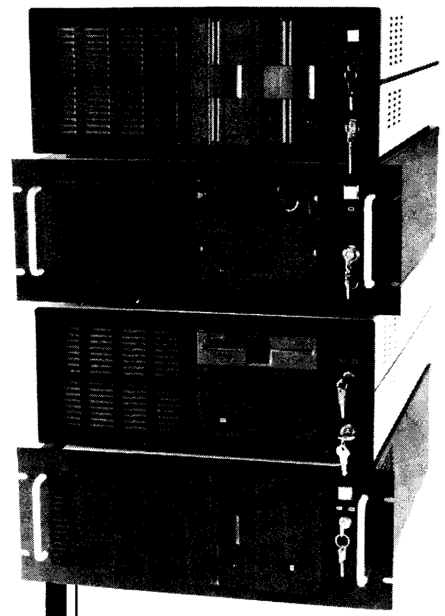
These routines were adapted from a good book on graphics, *High Performance Interactive Graphics: Modeling, Rendering and Animating For IBM PCs and Compatibles*, by Lee Adams. All the programs in the book are written in BASIC, yet I've found it an invaluable source of information on reflections, shading, and other similar topics.

Steven Byrnes
10510 Emnora
Houston, TX 77043

♦ ♦ ♦

Figure 3 — Quick EGA Screen Save/Restore

```
   void save_screen (char filename [13])
{
  char far *scrnptr;
  FILE *f;

  scrnptr = (char far *) 0xa0000000;  /* set up ptr to EGA mem */
  f = fopen (filename, "wb");      /* open file for binary write */

  outp (0x3ce, 4);  outp (0x3cf, 0);    /* set to read plane 0 */
  fwrite (scrnptr, 28000, 1, f);                /* save plane 0 */
  outp (0x3ce, 4);  outp (0x3cf, 1);    /* set to read plane 1 */
  fwrite (scrnptr, 28000, 1, f);                /* save plane 1 */
  outp (0x3ce, 4);  outp (0x3cf, 2);    /* set to read plane 2 */
  fwrite (scrnptr, 28000, 1, f);                /* save plane 2 */
  outp (0x3ce, 4);  outp (0x3cf, 3);    /* set to read plane 3 */
  fwrite (scrnptr, 28000, 1, f);                /* save plane 3 */
  outp (0x3ce, 4);  outp (0x3cf, 0);  /* reset to read plane 0 */
  fclose (f);
}  /* save_screen */


void get_screen (char filename [13])
{
  char far *scrnptr;
  FILE *f;

  scrnptr = (char far *) 0xa0000000;  /* set up ptr to EGA mem */
  f = fopen (filename, "rb");      /* open file for binary read */
  if (f==NULL) return;        /* if error opening file, return */

  outp (0x3c4, 2);  outp (0x3c5, 1);   /* set to write plane 0 */
  fread (scrnptr, 28000, 1, f);                  /* get plane 0 */
  outp (0x3c4, 2);  outp (0x3c5, 2);   /* set to write plane 1 */
  fread (scrnptr, 28000, 1, f);                  /* get plane 1 */
  outp (0x3c4, 2);  outp (0x3c5, 4);   /* set to write plane 2 */
  fread (scrnptr, 28000, 1, f);                  /* get plane 2 */
  outp (0x3c4, 2);  outp (0x3c5, 8);   /* set to write plane 3 */
  fread (scrnptr, 28000, 1, f);                  /* get plane 3 */
  outp (0x3c4, 2);  outp (0x3c5, 0xf); /*restore latch register*/
  fclose (f);
}  /* get_screen */

/***/
```



Integrand's new Chassis/System is not another IBM mechanical and electrical clone. An entirely fresh packaging design approach has been taken using modular construction. At present, over 40 optional *stock* modules allow you to customize our standard chassis to nearly any requirement. Integrand offers high quality, advanced design hardware along with applications and technical support *all at prices competitive with imports.* Why settle for less?

Rack & Desk PC/AT Chassis

- *Rack & Desk Models*
- *Accepts PC, XT, AT Motherboards and Passive Backplanes*
- *Doesn't Look Like IBM*
- *Rugged, Modular Construction*
- *Excellent Air Flow & Cooling*
- *Optional Card Cage Fan*
- *Designed to meet FCC*
- *204 Watt Supply, UL Recognized*
- *145W & 85W also available*
- *Reasonably Priced*

INTEGRAND RESEARCH CORP.

Call or write for descriptive brochure and prices:
8620 Roosevelt Ave. • Visalia, CA 93291
209/651-1203
TELEX 5106012830 (INTEGRAND UD)
EZLINK 62926572
We accept BankAmericard/VISA and MasterCard

IBM, PC, XT, AT trademarks of International Business Machines. Drives and computer boards not included.

Reader Service Number 22

Micro Ads

A Micro Ad is the inexpensive way to reach over 22,000 technical folks like yourself. To place a Micro Ad, just print out your message (make it short and sweet) and mail it to Micro C. We'll typeset your ad (no charge) and run it in the next available issue. You can also send camera ready copy. Rates: $99 for 1 time, $267 for three times, $474 for 6 times (a best buy at only $79 per insertion). Full payment must accompany ad. Each ad space is 2 1/4 inches by 1 3/4 inches.

OPT-TECH SORT/MERGE

Extremely fast Sort/Merge/Select utility. Run as an MS-DOS command or CALL as a subroutine. Supports most languages and filetypes including Btrieve and dBase. Unlimited filesizes, multiple keys and much more! MS-DOS **$149**. XENIX **$249**.

(702) 588-3737

Opt-Tech Data Processing

P.O. Box 678 - Zephyr Cove, NV 89448

Reader Service Number 64

RAM DISK

S-100

2 Meg, Port I/O

New, Warranteed

$775

S. Lugert
439 Peck Slip
NY, NY 10272

or call:
718-622-0654

Reader Service Number 52

Want to Throw Out your U.P.S. Log Book? Now You Can!

Here's what EASY-SHIP can do for you:

- ☑ Automatic U.P.S. Shipping to all of U.S & Canada.
- ☑ Fast, Easy Multiple-Shipments with All Options.
- ☑ U.P.S. Approved Shipping Labels & C.O.D. tags.
- ☑ Access to your ASCII Customer Data File.
- ☑ U.P.S. Approved Reports and Manifest Summary.
- ☑ *Approved Nationally by United Parcel Service.*
- ☑ NO MORE MANUAL LOGGING! And more!

For all IBM PC, AT, OS/2 Systems. **Only $365 + $3 S/H.**

Stat Supply Company
20214 Brondesbury, Katy, TX 77450
(800) 666-4567 or (713) 492-1931

68000 SOFTWARE

- K-OS ONE operating system uses MS-DOS disks with source code **$50**
- K-OS ONE manual **$10**
- HT68K SBC w/K-OS ONE. **$395**
- Screen Editor Toolkit **$50**
- HT-FORTH **$100**
- BASIC **$149**

Free newsletter & spec sheets

HAWTHORNE TECHNOLOGY

1411 S.E. 31st Ave., Portland, OR 97214
(503) 232-7332

Reader Service Number 34

8051
Z8 / Super8
C COMPILER

*** Call today for a FREE technical bulletin ***

MICRO COMPUTER CONTROL

P.O. Box 275 – Hopewell, NJ 08525 USA
Telex 9102404881 MICRO UQ

(609) 466-1751

Reader Service Number 100

ACCELERATE YOUR AT's MATH

Most ATs and clones run their 80287 math coprocessors at 4 to 6 Mhz. So if you have an 8 or 10 Mhz 80287 in your system, it's **loafing**! The Solution: Speed-up your AT's 80287's clock rate to it's maximum frequency with a coprocessor daughter-board. 8 or 10 Mhz versions available for $29.95 (80287 **not** included). Simple installation.

Sierra Circuit Design
18185 West Union Road
Portland, Oregon 97229
(503) 645-0734

TURBO C DEBUGGER

NO ASSEMBLY REQUIRED!

Trouble Shoot Your Programs From Your Source Code

- Window Style Interface
- Access Local & Global Variables by Name
- View Source Code
- Set Breakpoints
- Single Step & Trace
- Execution History

ONLY $35 Includes Tech Support & Printed Manual

DARKEL SOFTWORKS

Chatsworth, California (818) 341-9079

ONELINERS

for IBM PCs and compatibles

A pop-up (or command line) utility that instantly alphabetically lists both the current directory and the 1st line of all ASCII text files in a full screen window.
Automatically translates Wordstar and Wordperfect 4.2 files (including summary boxes) to text.
Indispensable for quickly identifying data, program source, word processor, etc. files from inside a running application.

ONLY **$15**

GNM Endeavors, Inc.
1910 Fieldwood Drive
Northbrook, IL 60062

FLASH

THE **DISK ACCELERATOR**

PC MAGAZINE EDITOR'S CHOICE

* EASY To Install
* Cache up to 3 MEGS of EXTENDED and or EXPANDED
* Buffers up to 26 DEVICE driven drives
* Comes with 2 FREE utilities!!!!!!!!

ORDER NOW ~~$89.95~~
(800) 25-FLASH **$19.95***

SOFTWARE MASTERS 6352 North Guilford Ave.
Indianapolis, In 46220 / 317) 253-8088

* To receive discount price, DEMAND IT!!
$5.00 shp/hnd in USA & CANADA, $15.00 overseas.

16 Megabytes EMS and/or Extended Memory

- Works on 8 or 16 bit bus
- 16 bit transfer on AT bus
- Single board design
- Includes RAM disk and extensive diagnostics
- Quantity/OEM discounts

XT and AT Compatible

Designed, Manufactured, Sold and Serviced by PTech

907 North 6th St. Lake City, MN 55041 (612)345-4555

Reader Service Number 54

snOOp - World's Best Disassembler. Turns any program to assembler source, comments each line of code. 8086 to 80386 processors and coprocessors. DOS calls, ports, EMS functions, topview calls and gets - all supported and unsupported codes. Batch or interactive. Built in code-sensitive help. Best way to: Learn Assembler, modify software when source is unavailable, find and disarm viruses, or just snOOp. $49.95 TriDOS
4004 SW Barbur • Portland, OR 97201
VISA/MC, AMEX, COD 800-237-9111

Reader Service Number 117

Get Inside Your Program Today!

Analyze and improve your program . . . function by function! Inside follows your program through execution counts, minimum, maximum, and total elapsed times with microsecond accuracy! Get inside your programs with unprecedented detail now!

Inside! Turbo C
Inside! Turbo Pascal
Inside! Quick C
Inside! Quick Basic
Inside! Microsoft Pascal
Inside! Microsoft Fortran
Inside! Lattice C
Inside! Logitech Modula-2

$75.00 Each

Visa/Mastercard Accepted

(800)537-5043

Paradigm Systems

P.O. Box 152 Milford, MA 01757

Reader Service Number 113

Is There A Gap In Your Info?

Fill in your Back Issues of Micro C today!

ISSUE #1 (8/81)
Power Supply
RAM Protection
Video Wiggle
1/2 PFM.PRN
16 pages

ISSUE #2 (10/81)
Parallel Print Driver
Drive Motor Control
Shugart Jumpers
Program Storage Above PFM
1/2 PFM.PRN
16 pages

ISSUE #3 (12/81)
4 MHz Mods
Configuring Modem 7
Safer Formatter
Reverse Video Cursor
FORTHwords Begins
16 pages

ISSUE #4 (2/82)
Keyboard Translation
More 4 MHz Mods
Modems, Lync, and S10s
Undoing CP/M ERASE
Keyboard Encoder
20 pages

ISSUE #5 (4/82)
Word Processing
Two Great Spells
Two Text Editors
Double Density Review
Scribble, A Formatter
20 pages

ISSUE #6 (6/82)
BBI EPROM Programmer
Customize Your Chars
Double Density Update
Terminal In FORTH
24 pages

ISSUE #7 (8/82)
6 Reviews Of C
Adding 6K Of RAM
Viewing 50 Hz
On Your Own Begins
24 pages

ISSUE #8 (10/82)
SOLD OUT

ISSUE #9 (12/82)
BBII EPROM Program
Relocating Your CP/M
Serial Print Driver
Big Board I Fixes
Bringing Up WordStar
Cheap RAM Disk
32 pages

ISSUE #10 (2/83)
SOLD OUT

ISSUE #11 (4/83)
SOLD OUT

ISSUE #12 (6/83)
256K for BBI
Bringing Up BBII
dBase II
Look at WordStar
Double Sided Drives for BBI
Packet Radio
5 MHz for Kaypro
40 pages

ISSUE #13 (8/83)
CP/M Disk Directory
More 256K for BBI
Mini Front Panel
Cheap Fast Modem
Nevada COBOL Review
BBI Printer Interface
Kaypro Reverse Video Mod
44 pages

ISSUE #14 (10/83)
BBII Installation
The Perfect Terminal
Interface To Electronic
Typewriter
BBI Video Size
Video Jitter Fix
Slicer Column Begins
Kaypro Color Graphics Review
48 pages

ISSUE #15 (12/83)
Screen Dump Listing
Fixing Serial Ports
Playing Adventure
SBASIC Column Begins
Upgrading Kaypro II To 4
Upgrading Kaypro 4 To 8
48 pages

ISSUE #16 (2/84)
Xerox 820 Column Restarts
BBI Double Density
BBII 5"/8" Interface Fix
Kaypro ZCPR Patch
Adding Joystick To Color
Graphics
Recovering Text From Memory
52 pages

ISSUE #17 (4/84)
Voice Synthesizer
820 RAM Disk
Kaypro Morse Code Interface
68000-Based System Review
Inside CP/M 86
56 pages

ISSUE #18 (6/84)
Kaypro EPROM Programmer
I/O Byte: A Primer
Kaypro Joystick
Serial To Parallel Interface
Business COBOL
60 pages

ISSUE #19 (8/84)
Adding Winchester To BBII
6 MHz On The BBI
Bulletin Boards
Track Buffering On Slicer
4 MHz For The 820-I
64 pages

ISSUE #20 (10/84)
HSC 68000 Co-Processor
DynaDisk For The BBII
Serial Printer On BBI Sans S10
Cheap & Dirty Talker For Kaypro
Extended 8" Single Density
72 pages

ISSUE #21 (12/84)
Analog To Digital Interface
Installing Turbo Pascal
Low Intensity BBI Video
Turbo Pascal, The Early Days
80 pages

ISSUE #22 (2/85)
Xerox 820-II To A Kaypro-8
Sound Generator For the
STD Bus
Reviews Of 256K
RAM Expansion
In the Public Domain Begins
88 pages

ISSUE #23 (4/85)
Automatic Disk Relogging
Interrupt Drive Serial Printer
Low Cost EPROM Eraser
Smart Video Controller
Review: MicroSphere RAM Disk
Future Tense Begins
86 pages

ISSUE #24 (6/85)
C'ing Into Turbo Pascal
8" Drives On The Kaypro
48 Lines On A BBI
68000 Versus 80x86
Soldering: The First Steps
88 pages

ISSUE #25 (8/85)
Why I Wrote A Debugger
The 32-Bit Super Chips
Programming The 32032
Modula II
RS-232C: The Interface
104 pages

ISSUE #26 (10/85)
Inside ZCPR3
Two Megabytes On DSI-32
SOG IV
The Future Of Computing
Graphics In Turbo Pascal
104 pages

ISSUE #27 (12/85)
SOLD OUT

ISSUE #28 (2/86)
Pascal Runoff Winners
Rescuing Lost Text From
Memory
Introduction To Modula-2
First Look At Amiga
Inside The PC
104 pages

ISSUE #29 (4/86)
Speeding Up Your XT
Importing Systems
From Taiwan
Prototyping In C
C Interpreters Reviewed
Benchmarking The PCs
104 pages

ISSUE #30 (6/86)
PROLOG On The PC
Expert Systems
Logic Programming
Building Your Own Logic
Analyzer
256K RAM For Your 83 Kaypro
PC-DOS For Non-Clones
104 pages

ISSUE #31 (8/86)
RAM Resident PC Speedup
Practical Programming In
Modula-2
Unblinking The PC's Blinkin'
Cursor
Game Theory In PROLOG
and C
104 pages

ISSUE #32 (10/86)
Public Domain 32000:
Hardware And Software
Writing A Printer Driver for
MS-DOS
Recover A Directory By
Reading & Writing Disk
Sectors
96 pages

ISSUE #33 (12/86)
SOLD OUT

ISSUE #34 (2/87)
SOLD OUT

ISSUE #35 (4/87)
SOLD OUT

ISSUE #36 (6/87)
Mouse Control
Build A Midi Interface
For Your PC
Designing A Database, Part 2
Interrupts On The PC
Hacker's View of MS-DOS
Vs 3.X
Digital To Analog Conversion,
A Designer's View
96 pages

ISSUE #37 (9/87)
Desktop Publishing On A PC
Build Your Own Hi-Res Graphics
Scanner For $6, Part 1
Designing A Database, Part 3
Controlling AC Power
From Your PC
Expanded Memory On The
PC/XT/AT
Uninterruptable Power
Supply For RAM Disks
96 pages

ISSUE #38 (11/87)
Parallel Processing
Laser Printers, Typesetters
And Page Definition
Languages
Magic In The Real World
Build A Graphics Scanner
For $6, Part 2
Writing A Resident Program
Extractor In C
96 pages

ISSUE #39 (1/88)
PC Graphics
Drawing The Mandelbrot And
Julia Sets
Desktop Graphics
Designing A PC Work-
station Board
Around the TMS-34010
96 pages

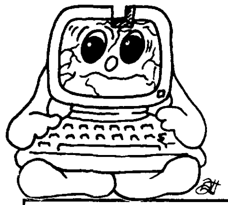
ISSUE #40 (3/88)
The Great C Issue
11 C Compilers
Writing A Simple Parser In C
C++, An Object Oriented C
Source Level Debugger For
Turbo C
96 pages

ISSUE #41 (5/88)
Artificial Intelligence
3-D Graphics
Neural Networks
Logic Of Programming
Languages
Applying Information Theory
96 pages

ISSUE # 42 (6/88)
Maintaining PCs
Keeping Your Hard Drives
Running
Troubleshooting PCs
XT Theory of Operation
Simulating A Bus
Ray Tracing
96 pages

ISSUE #43 (9/87)
Building Databases
Build a C Database
Selecting a dBase III
Compatible Compiler
Working with Paradox
Designing Custom PC Cards
Accessing dBase III Plus
Records from Turbo Pascal
96 pages

♦♦♦



To Order:

Phone: **1-800-888-8087**
Mail: **PO Box 223**
Bend, Oregon 97709

United States,	
Issues #1-34	$3.00 each ppd.
Issues #35-current	$3.95 each ppd.
Canada, & Mexico	
All issues	$5.00 each ppd.
Foreign (air mail)	
All Issues	$7.00 each ppd.

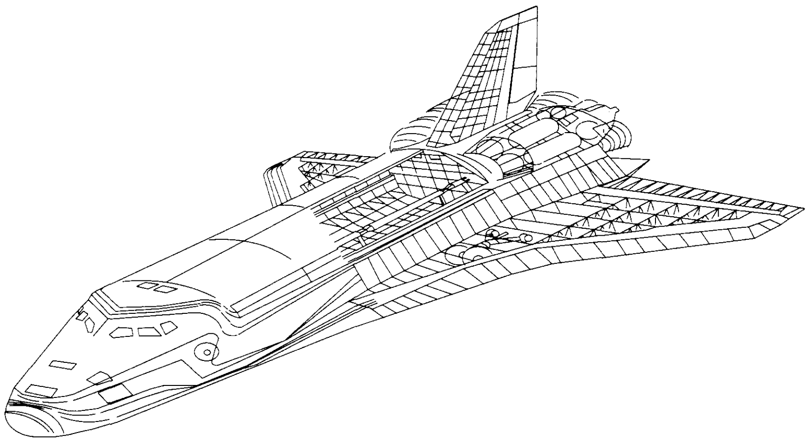
ADVERTISERS INDEX

Issue 44

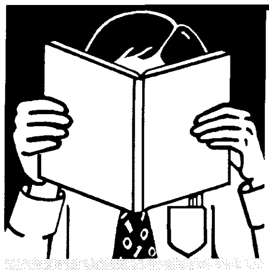
Reader Service		Page Number
72	Acquired Intelligence	24
107	American Cosmotron	49
**	Andsor Research	63
04	Austin Codeworks	61
15	Cascade Electronics	37
119	CC Software	75
118	CNS	68
71	Complete Logic Systems	29
105	Computerized Processing Unltd.	31
07	CompuView	Inside Front Cover
90	Dair Computers	53
32	Digital Research	52
111	Digitalk	12,13
16	Dreamtech	86
09	Ecosoft	11
10	Emerald Microware	57
93	Erac Company	7
112	Garrison, Peter	17
11	Halted Specialties	41
22	Integrand	92
**	JOOP	23
88	Koala Computers	25
12	Logitech Inc	34,35
17	Manx Software Systems	69
103	Max Software	70
42	McTek Systems	59
**	Micro Cornucopia	89,94
37	Microprocessors Unltd	53
24	microSOLUTIONS	15
02	Microsphere	01
59	National Advancement Corp	71
110	NuMega	2
85	Oregon Software	71
03	PC Tech	Cover
119	Peripheral	73
109	Quarterdeck	5
60	Scientific Software Solutions	52
108	Sofsolutions	28
19	Slicer Control/Computer	64
40	Star-K Software Systems	44
101	United Products	91
62	V Communications	82
116	Whitewater Group	21
39	Xenosoft	72
70	Zortech, Inc.	Inside Back Cover

** Contact Advertiser Directly.

Coming in Issue #45 — CAD



- **Electronic Circuit CAD Systems**
- **State of the Art Object Oriented Graphics (CAD)**
- **Plus: Secrets of Compiler Optimization**
- **And: Interrupt Service Routines In C**



THE LAST PAGE

By Gary Entsminger
1912 Haussler Dr.
Davis, CA 95616

Reasonable Dreams

Gary's off on another tangent. This one's as simple or as complex as you want to make it.

In his new book, *The Dreams of Reason*, physicist Heinz Pagels calls the computer, "the instrument of complexity." Computers allow physicists, chemists, biologists, economists, social scientists, and others to explore (and sometimes solve) complex problems they couldn't tackle before.

"Computers, because of their capacity to manage enormous amounts of information, are showing us new aspects of social reality ... The power of the computer (for research) lies in its capacity to computationally model and simulate complex systems."

We've already begun to explore several of these major avenues of complexity in *Micro C* ("Fractals, etc.," in issues #39 and #43; "Chaos, Butterflies & The BGI," issue #42), and I'm pleased with Pagels' attempt to define this new science of complexity.

Complexity, as you might expect, is an illusive concept, and like anything new suffers (and benefits) from its "newness."

What Is Complexity?

In our ordinary language, "complexity" refers to a state or condition consisting of many interacting components. The interaction of the components is complicated; in other words, we probably can't easily distinguish all the interactions at once.

Although this kind of general description is accurate, it's hardly "scientific," and in fact isn't very useful. So Pagels suggests a more quantitative, less qualitative definition —

"Complexity is a quantitative measure that we can assign to a physical system or a computation that lies midway between the measure of simple order and complete chaos."

A diamond, he goes on to say, has neatly arranged atoms, and is therefore ordered. A rose consists of randomness and order in the arrangement of its parts, so it's complex. The movement of gas molecules is chaotic.

Randomness

Since we're looking for a quantitative measure of complexity, perhaps the simplest first approach we can take is to examine numbers, in general.

What is it about a number, for example, that makes it "random" or "not random." The number —

```
.010101010101
```

for example, seems ordered. And at least it seems we can predict the next digit (0 or 1), depending on the previous digit.

A number like —

```
.18976454329657345292467
```

however, seems disordered.

What we'd like to do is quantify the distinction between these (and other) numbers. One approach we can take is to examine their "computability." In other words, what kind of algorithm would generate the particular number?

Consider another example —

```
.7142857142...
```

Although this number looks complicated (or random), it's actually easy to generate. It's simply 5 divided by 7. Or as an algorithm: divide 5 by 7 and write the result.

Offhand, I know of no algorithm generate the earlier number —

```
.18976454329657345292467
```

other than something like: write .18976454329657345292467. Which explicitly specifies the number within the algorithm. In other words, the number isn't "computable" from other numbers.

Turing and others have used this non-computability as a possible definition of randomness. "For computable numbers, even if they're infinitely long, it's possible to write a relatively short program that will calculate them. For the non-computable (random) numbers, the only algorithm that will do the job already contains all the information in the number explicitly — the algorithm is at least as long as the number."

From this, we might then take a next logical step and compute a number's complexity from the length of its algorithm.

Challenging You

Over the next few issues, I'd like your help in exploring and quantifying complexity. For reasons I'll save until next time, I'm not convinced that algorithmic length is complexity's key quantifier.

What do you think? You might begin by sending me your definition of randomness. If you can, quantify it (via an algorithm or program).

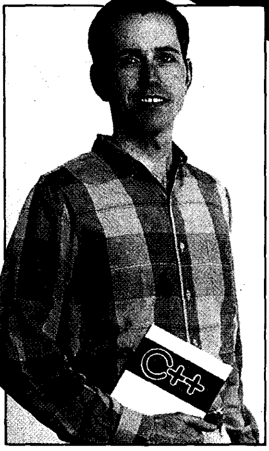
Meantime, check out Pagels' fascinating book, *The Dreams Of Reason* (from Simon & Schuster) — and send me your quantifiable dreams, reasonable or otherwise.

♦ ♦ ♦

INTRODUCTORY PRICE

$99.95

Regular price $149.95.



WALTER BRIGHT – Programmer

"This is a revolutionary product which reflects the fact that Walter Bright is one of the world's best programmers."
Andy Redfern, Personal Computer World

"I find it amazing that Walter continues to find ways to improve his already excellent product."
Scott Robert Ladd, Micro Cornucopia

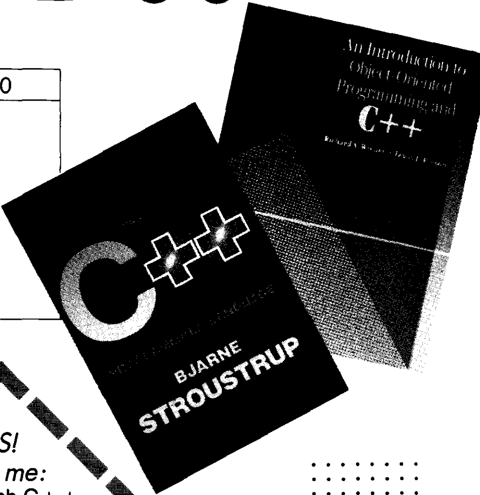
C is great! C++ is better!!

Performance benchmarks

Benchmark	Zortech C	Zortech C++	Turbo C 1.5	Quick C 1.0
Sieve	20.49	20.54	23.62	22.72
Rsieve	20.49	20.54	23.62	22.03
Integer	1.32	1.38	6.31	6.49
Float	0.17	0.22	52.29	51.03
*Float	32.73	37.74	52.39	51.63
Pointer	17.91	17.96	17.13	16.87
Rpointer	17.79	17.91	17.14	16.64
Loop	3.90	3.90	3.90	3.90
Optimize	0.49	0.60	8.46	8.79

Benchmarks were run on an 80286 based IBM compatible at 6Mhz with no 8087.

*The float benchmark was re-run without optimization.



Zortech C++ is the first native-code C++ compiler for MS-DOS compatible machines. This means you don't need to spend a lot of money on a C compiler. In fact, Zortech C++ comes with a C++ compiler, a C compiler, a linker, a librarian, a fully integrated editing environment, context sensitive help and the fastest graphics library you've ever seen!

With Zortech C++, you can mix and match code in C++, C and assembler and link them all together with a single command!

Zortech C++ comes with a complete C++ and C library, including compatibility with the ANSI C draft. Unique to Zortech C++ is support for mice, fast screen writes, re-entrant floating point and speaker support.

Now you can move your Microsoft and Turbo C code over to Zortech C++ – we've included compatible library functions! Also, we have made Zortech C++ compatible with Microsoft's Codeview debugger!

Enter the world of Object Oriented Programming (oops) with Zortech C++!

Sure, C is great – but C++ is better!

NOW AVAILABLE...
Std. Library Source Code
*Only $49.95 * (with C++ Purchase)*

YES!
Rush me:

☐ Zortech C++ 99.95*
☐ Library Source $49.95*
☐ Intro to OOP and C++ $27.95*
☐ The C++ Programming Language $29.95

VISA/MC/COD CHECK ACCEPTED

Name..
Address..
.... *Phone*..............
VISA or MC#.............................. *Exp. date*..............

ZORTECH

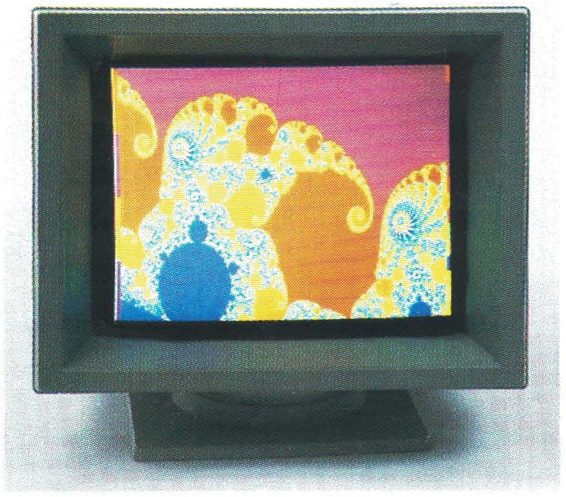
To: ZORTECH INC. 366 Massachusetts Ave., Arlington, MA 02174. Tel: 617-646-6703. Fax: 617-643-7969.

CALL THE ORDER HOTLINE 1-800-848-8408

* Please add $5.05 shipping to your order (UPS Ground)

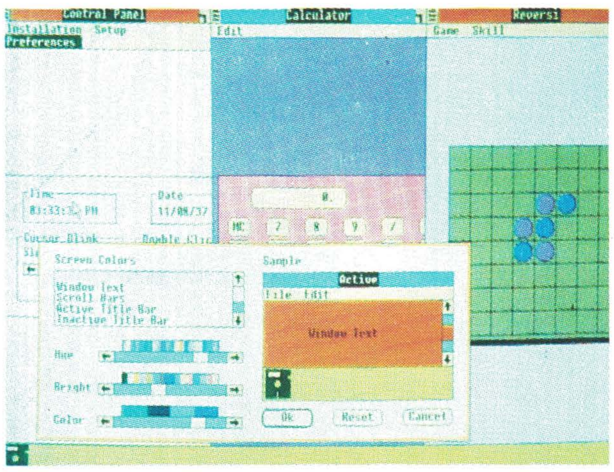
VERY HIGH RESOLUTION

The PC Tech COLOR and MONOCHROME video processor boards employ the TMS 34010 high performance graphics co-processor to insure the best possible video performance at reasonable prices.



Color 34010 Video Processor:

- Featured on the cover of Micro Cornucopia.
- From 800 x 512 through 1024 x 800 resolution (depending on monitor and configuration).
- 8 Bits per pixel for 256 simultaneous colors
- Hardware support for CGA/MDA emulation.
- PC, XT, and AT compatible



The PC Tech Color 34010 video processor is a superior 34010 native code and DGIS development tool. We support up to 4 megabytes of program (non-display) 34010 RAM as well as up to 768K bytes of display RAM. ***Compare our architecture and prices to any other intelligent graphics board. Then choose the PC Tech Color 34010 Video Processor for your development engine and your production requirements as well.***

Color 34010 Video Processor $1,350.00

Price includes 512K display RAM, 1024K program RAM, and utility software. Monitor not included.
Also available: DGIS, 34010 C compiler, assembler, 34010 fractal software, additional display and program memory, and various monitor options.

PC Tech Monochrome 34010 Video Processor and Monitor

- 736 x 1024 resolution (other options available)
- 2 bits per pixel for 4 hardware gray shades
- Hardware support for CGA/MDA/Hercules emulation
- PC, XT, and AT compatible
- Full page 66 line text editing with many popular editors
- Excellent windows 2.0 application development system

The graphics and bit manipulation capabilities of the TMS 34010 make the PC Tech Monochrome 34010 Video Processor 66 line full page text and graphics display faster than many 25 line systems. The video processor is available separately or with the high resolution white phosphor monitor shown above.

Monochrome 34010 Video Sub-System $1,495.00

Price includes Monochrome Video Processor and monitor pictured above.
Also available: DGIS, TI 34010 C compiler, TI assembler.
Monochrome 34010 Video Processor also available separately.

Designed, Sold and Serviced By:

PC Tech

**904 N. 6th St.
Lake City, MN 55041
(612) 345-4555
(612) 345-5514 (FAX)**

Reader Service Number 3

PC, XT, AT, DGIS, Hercules, and Windows 2.0 are trademarks or registered trademarks of their respective companies.