August-September 1986

Practical Projects

New PC Speedup With RAM Resident Switch ............ 6
A cheaper, fancier, two-speed clock that you control from the keyboard.

Practical Programming In Modula-2 ...................... 12
A window lover's dream, right down to the last frame.

Unblinking The PC's Blinkin' Cursor ..................... 22
Tired of that irritating flashing cursor? Here's the fix.

Game Theory Models in PROLOG and C .................. 24
A modeling system that ties Turbo PROLOG together with Microsoft C.

Serial Communications Program For CP/M & MS-DOS ... 30
Tying two aliens together with Turbo pascal.

Eight Booleans In A Byte ............................... 58
A bit pickers special feature.

Hard Drives: The Service Perspective ................... 62
Which manufacturers support the purchaser? Which don't?

Purchasing And Installing A Hard Drive .................. 66
Join us for a quick read between the tracks.
Borland introduces Turbo Prolog, the natural language of Artificial Intelligence.

Prolog is probably the most powerful computer programming language ever conceived, which is why we've made it our second language—and "turbocharged" it to create Turbo Prolog.

Our new Turbo Prolog brings supercomputer power to your IBM® PC and introduces you step-by-step to the fascinating new world of Artificial Intelligence. And does all this for an astounding $99.95.

Even if you've never programmed before, our free tutorial will get you started right away.

You'll get started right away because we have included a complete step-by-step tutorial as part of the 200-page Turbo Prolog Reference Manual. Our tutorial will take you by the hand and teach you everything you're likely to need to know about Turbo Prolog and artificial intelligence. For example, once you've completed the tutorial, you'll be able to design your own expert systems utilizing Turbo Prolog's powerful problem-solving capabilities.

Think of Turbo Prolog as a high-speed electronic detective. First, you feed it information and teach it rules. Then Turbo Prolog "thinks" the problem through and comes up with all the reasonable answers—almost instantly.

If you think that this is amazing, you just need to remember that Turbo Prolog is a 6th-generation language—and the kind of language that 21st-century computers will use routinely. In fact, you can compare Turbo Prolog to Turbo Pascal by the way you could compare Turbo Pascal to machine language.

You get the complete Turbo Prolog programming system for only $99.95.

You get a complete Turbo Prolog development system including:

- The lightning-fast Turbo Prolog incremental compiler and the interactive Turbo Prolog editor.
- The 200-page reference manual which includes the step-by-step Turbo Prolog tutorial.
- The free GeoBase® natural language database including 3000 artificial intelligence concepts, in-disk-ready to compile. GeoBase is a complete database designed and developed around U.S. geography. It includes cities, mountains, rivers, and highways, and comes complete with natural language use. GeoBase immediately "as is," or modify it to fit your own interests.

And don't delay—don't waste a second—get Turbo Prolog now. $99.95 is an amazingly small price to pay to become an immediate authority—an instant expert on artificial intelligence! The 21st century is only one phone call away.

**YES!** I want the best Turbo Prolog at only: **$99.95**

To order by phone, or for a dealer nearest you, Call (800) 255-8008 in CA call (800) 742-1133.

Send me Turbo Prolog at $... Outside USA add $10 per copy and add res. applicable sales tax for...

Amount enclosed:

Credit card expiration date:

Your order now includes:

- Turbo Prolog manuals:
  - Turbo Prolog manual
  - Turbo Prolog tutorial manual
  - Turbo Prolog manual

The total cost is: $139.95

**NOT COPY PROTECTED**

60-DAY MONEY-BACK GUARANTEE

Call 800-255-8008 or your dealer

Order now and get Turbo Pascal for only $49.95.

Borland International, 3385 Scotts Valley Drive, Scotts Valley, CA 95066

Borland International is a registered trademark of Borland International, Inc. Turbo Pascal is a registered trademark of Borland International, Inc. IBM, PC, XT, AT, PCjr, EBCDIC, and EGA are registered trademarks of International Business Machines Corporation. Copyright 1988 Borland International. 09-005-00044

TRAVEL IMMEDIATELY Turbo Pascal is to Prolog what Turbo Pascal is to Pascal. Our Turbo Pascal astonished everyone who thought of Pascal as just another language. We changed all that—and now Turbo Pascal is the de facto worldwide standard, with hundreds of thousands of enthusiasts and users in universities, research centers, schools, and with professional programmers, students, and hobbyists.

You can expect at least the same impact from Turbo Prolog, because while Turbo Prolog is the most revolutionary and natural programming language, it is also a complete development environment—just like Turbo Pascal.

Turbo Prolog radically alters and dramatically improves the brave new world of artificial intelligence—and invites you into that fascinating universe for a humainly intelligent $99.95.
**DISK DRIVES**

- 3.5" MMI HARD DISK .......... $249.95
  - 10 MEG
  - ST212/412 COMPATIBLE
  - SHOCK MOUNTED IN 1/4 HEIGHT
- 5 1/4" FRAME WIBEZEL

**PACKAGE DEAL** — ABOVE HARD DISK WITH OMTI IBM CONTROLLER CARD, CABLES AND FORMATTED READY TO PLUG IN YOUR IBM OR COMPATIBLE JUST $399.00

- 3.5" EPSON SMD 100 .......... $89.00
  - DOUBLE SIDED, 1 MB UNFORMATTED
  - HARDWARE COMPATIBLE WITH 5 1/4"
  - 96 TPI DRIVES
  - WITHOUT FRONT PANEL

**HALTED. SPECIALTIES** offers unique supermarket style shopping for your electronic needs. We stock thousands of parts, from the newest IC's to some of the first transistors. We also have computer accessories, test equipment, tools, R&D supplies and much more. Please call or visit one of our retail stores.

<table>
<thead>
<tr>
<th>Item Description</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 1/4&quot; ST506 HARD DISK</td>
<td>$79.95</td>
</tr>
<tr>
<td>5MEG</td>
<td></td>
</tr>
<tr>
<td>153 CYLINDER, 4 HEAD</td>
<td></td>
</tr>
<tr>
<td>5 1/4&quot; TEC FB504</td>
<td>$95.00</td>
</tr>
<tr>
<td>96 TPI HALF HEIGHT</td>
<td></td>
</tr>
<tr>
<td>720 KB FORMATTED CAPACITY</td>
<td></td>
</tr>
<tr>
<td>QUIET!</td>
<td></td>
</tr>
<tr>
<td>RUNS ON IBM XT WITH J FORMAT</td>
<td></td>
</tr>
<tr>
<td>SOFTWARE (NOT SUPPLIED)</td>
<td></td>
</tr>
<tr>
<td>8&quot; SHUGART SA801R</td>
<td>$179.00</td>
</tr>
<tr>
<td>FULL HEIGHT</td>
<td></td>
</tr>
<tr>
<td>SINGLE SIDED</td>
<td></td>
</tr>
<tr>
<td>600 KB FORMATTED CAPACITY</td>
<td></td>
</tr>
<tr>
<td>BRAND NEW! NOT RECONDITIONED</td>
<td></td>
</tr>
<tr>
<td>XEBEC S1410A HARD DISK CONTROLLER</td>
<td>$149.95</td>
</tr>
<tr>
<td>REQUIRES SASI HOST ADAPTOR</td>
<td></td>
</tr>
<tr>
<td>LAMDA POWER SUPPLY CLOSEOUT</td>
<td></td>
</tr>
<tr>
<td>LXS SERIES</td>
<td></td>
</tr>
<tr>
<td>+ 12 VDC @ 10.5 AMP</td>
<td>$29.95</td>
</tr>
<tr>
<td>+ 5 VDC @ 27.5 AMP</td>
<td>$34.95</td>
</tr>
<tr>
<td>+ 5 VDC @ 9.0 AMP</td>
<td>$24.95</td>
</tr>
<tr>
<td>RELAY</td>
<td></td>
</tr>
<tr>
<td>12 VDC</td>
<td></td>
</tr>
<tr>
<td>SPDT</td>
<td></td>
</tr>
<tr>
<td>5 AMP</td>
<td></td>
</tr>
<tr>
<td>75°</td>
<td></td>
</tr>
<tr>
<td>TRANSFORMER</td>
<td></td>
</tr>
<tr>
<td>18 VAC</td>
<td></td>
</tr>
<tr>
<td>.75 AMP</td>
<td></td>
</tr>
<tr>
<td>50°</td>
<td></td>
</tr>
</tbody>
</table>

**SPECIAL OF THE MONTH**

5 MEGABYTES FOR YOUR IBM OR COMPATIBLE

- IBM* COMPATIBLES
  - All of our IBM* Compatible Products were carefully tested and chosen for full compatibility.
  - FLIP TOP CASE ............. $49.95
  - 130 W POWER SUPPLY ........ $89.95
  - 150 W POWER SUPPLY ........ $99.00
  - KEYBOARD DKM 2000 .......... $79.95
  - KEYBOARD KB 5151 (Enhanced) $99.00
  - 640K XT* MOTHERBOARD (OK RAM) $129.95
  - BIOS ROM (SIDEKICK COMPATIBLE) $29.95
  - MULTI I/O W/FDD CONT ...... $129.95
  - COLOR GRAPHICS CARD ....... $99.95
  - MONOCHROME CARD ........... $119.95
  - MULTIFUNCTION CARD ....... $129.95
  - FLOPPY CONTROLLER .......... $49.95
  - HARD DISK CONTROLLER ...... $175.00
  - INTERNAL 1200 BAUD MODEM ... $199.00
  - 2MB RAM EXPANSION FOR X* .. $199.00
  - 3MB MULTIFUNCTION CARD FOR IBM* WITH OK RAM $225.00
  - EPROM PROGRAMMER (27xxx) .. $129.95
  - EGA CARD (IBM* COMPATIBLE) $295.00
  - *XT, AT AND IBM are Registered Trademarks of International Business Machines Corp.

**ACCESSORIES**

- Parallel printer cable ........ $ 9.95
- Swivel monitor stand .......... $12.95
- Y power cable for disk drive ... $ 2.95
- NEC V20 processor 8mhz. (8088) $19.95
- NEC V20 processor 8mhz. (8088) $24.95
- 4164 150 ns 64K ram chips .... $ 1.19
- 41256 150 ns 256K ram chips .... $ 2.99
- NEC V30 processor 8mhz. (8086) $ 25.00
- 8087 math coprocessor ...... $139.00

**HALTED SPECIALTIES CO., INC.**

827 E. EVELYN AVE., SUNNYVILLE, CA 94086

MAIL ORDERS CALL: (408) 732-1573

3 CONVENIENT LOCATIONS

HSC Electronic Supply of Santa Rosa
6819 S. Santa Rosa Ave.
Cotati, CA
(707) 792-2357

HSC Electronic Supply
5549 Hemlock Street
Sacramento, CA
(916) 338-2545

**TERMS:** Minimum order $10.00. California residents add 7% sales tax. Prepaid orders sent freight C.O.D. or call for changes, shipping will be added to credit card and C.O.D. orders. Prepaid orders over $100.00 use money order or certified check. Please do not send cash. Some items limited to stock on hand. Prices subject to change.
EDITORIAL

Tongue Tied

Well Said

"Hurry up and decide what you're going to talk about at SOG."

Gary was pushing me for a title to my SOG V talk. Last year I was more an interested observer than a participating speaker.

"SOG is the place where specialists stick their heads up and see what other people are doing. It also gives them a chance to speak about their own projects — about what they're really into," he continued.

It's easy for them to talk, I thought. What have I been doing? What am I excited about? What kind of unique information do I have? Good question.

I've traded my engineering bench for all the excitement of publishing. I've learned a lot about hiring and firing, finances, printers, typesetters, Kaypros, columnist, large mailing lists, running technical forums, and maintaining an office sprinkler system. (The grass really is greener on the neighbor's side.)

Fortunately, I've come into contact with some really super people who haven't traded in their benches for broader hats. People who have concentrated on one thing. They are the people who should (and will) speak at SOG.

Not that I mind speaking ... It's a lot like writing except that people can hiss and throw things in real time. But subject is another question.

If I talk about what I'm really working on (defective sprinkler valves) maybe Gary won't be so insistent next year.

Quiet

There is something very fragile about writing. It's a discussion with an imaginary someone — recorded for probable deletion.

In my case, this special conversation is very much affected by the situation. If I feel hassled, harried, then my writing gets choppy, curt. I become very literal because my mind pictures aren't working. This kind of copy isn't fun to read or edit.

When things are very quiet, very relaxed, and I'm not distracted by those little things that seem so important during the 8 to 5 office workout, words can take on a life, a collection of colors that make them an art form unmatched by any other.

It's A Racket

Unfortunately, one of the most constant things about computers, especially the latest, greatest, fanciest, and fastest, is that they are noisy. Those fancy little 135 watt supplies have buzzy little fans. Winchesters wind up like kids' tops.
Features

6  Larry Fogg  New PC Speed-up With RAM Resident Soft Switch  
   We thought we'd covered everything there was to cover about speeding up a PC...

12  James Albert  Practical Programming In Modula-2: Controlling The Video On PCs  
    Fun with Wirth's Latest language.

22  Sigi Kluger  Unblinking The PC's Blinkin' Cursor  
    If your blinking cursor is keeping you awake...

24  Gary Entsminger  Game Theory Models In PROLOG and C  
    A heavy duty combination of two great languages.

30  Greg Flothe  Serial Communications Program For CP/M And MS-DOS  
    If you have the connections, we have the code.

58  William Meacham  Eight Booleans In A Byte: Operations In Turbo Pascal  
    Down to picking bits in Turbo Pascal.

62  Rebecca Ozrelic  Hard Drives: The Service Perspective  
    Who's providing support after the sale, the seller or the manufacturer?

66  David Thompson  Purchasing And Installing A Hard Drive  
    We found lots of information about hard drives. Most of it wasn't very useful.

74  Doug Anderson  Poor Man's Networking: Connecting Two CP/M Machines  
    Review of a simple, software only, local area network.

82  Don Fletcher  Build Your Own Inexpensive Logic Analyzer: Part 2, The Software  
    The response to this project has been super.

88  Dave Rand  The Green Hills C Compiler  
    Dave looks at a highly optimized C compiler for the 32032.

Columns

40  86 World  How to run programs from other programs.

48  On Your Own  A fast, clean solution to dBASE III.

52  Pascal Procedures  Philip Hess covers run time allocation of static arrays in Turbo Pascal.

54  Kaypro  An inside look at what makes the 84's video drop characters.

70  In The Public Domain  Steve talks about access to languages, the usual and the unusual.

72  The S-100 Bus  Z80 slave boards highlight this column.

84  C'ing Clearly  This small model C can address lots of memory.

95  Culture Corner  6th annual crossword puzzle celebrates our 5th anniversary issue.

98  Technical Tips

Future Tense  By Gary Entsminger

96  Tidbits  Gary looks at a super bargain statistical package and learns to eat worms.

104  The Last Page  Seeding the random number generator. (Without growing daisies.)
Article Update

I received a letter from a reader requesting clarification of the part number of the transformer in my article “Monitoring A Modem With A Bridge” from Micro C issue #29. I find that Radio Shack no longer sells that part. I’ve done a quick test based on another part and find this can be used as an electrical substitute, though it may not fit in the pipe.

The part is a small 1000 to 8 ohm transformer (part #273-1380). The turns ratio will be about 11:1. I changed the 10K resistor to 2.2K and the 100 ohm resistor to 150 ohms, and overall voltage reduction was still okay, about 175:1. Actually, any transformer with a rated primary impedance of 500 to 2000 ohms and a secondary with equal or somewhat less impedance than the primary will work, as long as the turns ratio is taken into account. The overall reduction should be between 100 and 200:1. The turns ratio is the square root of the impedance ratio. The reduction ratio is the turns ratio times the ratio of the resistor, i.e., \( \frac{5q-Rt(1000/8)}{(2200 + 150)} = 175.1 : 1 \) for this example.

Philip Elrod
3245 Spring Dr. NW
Doraville GA 30360

User’s Guide Returns?

We were surprised to see mention of "User’s Guide" in your editorial in issue #29. While it’s true that "User’s Guide" stopped publishing last November, and "PC User" never started, we have not disappeared. Both magazines were bought by Mach Gelt, Inc. (MGI) of New York, which has agreed to fulfill all paid subscriptions up to their expiration dates with a new magazine, slated to debut between July and September of this year. The magazine will cover CP/M, and we’ll be writing for it, so subscribers will get something very similar to what they paid for.

We hope this new arrangement satisfies our subscribers.

Tony Bove and Cheryl Rhodes
Founders & Editors of "User’s Guide"
P.O. Box 5245
Redwood City CA 94063

Wants Modula-2 Update

I read with great interest in Micro C issue #25 excerpts from a talk by Niklaus Wirth on Modula-2 in which he described his current work on a port of Modula-2 to the National Semiconductor 32000 family of CPU clusters. The match of such an elegant hardware implementation with the elegance and simplicity of Modula-2 seems an ideal partnership. However, I haven’t seen or heard anything since that article. Do you know how you could find out more about this project?

Randy George
680 Harlan
Lakewood CO 80214

Editor’s note:

Hoo boy! Wirth was working at Xerox Parc in Palo Alto, California (that’s the place where the Mac interface was developed). I haven’t heard anything out of Xerox lately, now that you mention it. Maybe they’re waiting for someone to leave the organization and start a new company based on the research.

Cheap Floppies

Here’s a tip if you use lots of floppy disks. Unitech charges 85 cents per disk for double sided disks boxed in lots of ten, labeled and with plain envelopes. I’ve been using them for several weeks and the quality seems very good. Their minimum order is $17 plus $3 shipping. (I used to buy in bulk from Spite Software, but they are slow and the minimum order is 100.) Unitech’s address is 20 Hurley St., Cambridge, MA 02141.

Henry C. Davis
Box 532, 207 E. Market St.
Georgetown DE 19947

Micro C Kaypro Manual?

I have a sure-fire winning idea for Micro C. Why not compile all the letters, articles, Kaypro Columns, Technical Tips, etc. which have anything to do with the CP/M Kaypro? You could put these all together in one booklet and maybe add some new technical material to fill in the gaps. Then you could publish and sell this booklet. I’m positive it would be one of your biggest selling products. It wouldn’t take that much work, since the bulk of the material is already written.

Walker Kennedy III
949 E. 800 S., Apt. 11
Salt Lake City UT 84102

Editor’s note:

Heck of an idea, Walker. I think it would be a good seller. A year ago we were definitely thinking about doing this very thing. However, we decided to spend our time looking forward — learning, developing new contacts, changing the look and editorial direction of Micro C. You can see the results of that now, but because of that a lot of really good ideas have been put aside.

Poor Man’s CompuServe

The Dorsai Embassy is an RBBS in New York City which caters to all micros and offers free public access. The system is standard XModem, Christensen protocol, and offers storyboarding and conferencing. To log on, type : and then 'help'. The bulletin board will tell you what to do next. Users have limited access until they register. Dorsai is currently negotiating with SIG/M and PC/Blue to put their internal library on the RBBS.

Charles Rawls, Sysop
41 Crosby St.
New York NY 10013
(212) 226-2394
(212) 966-6406
(212) 431-1944
(212) 219-9840

Expansion Bay For A Tandy

I’ve been thinking of adding a 20 Meg hard disk in a separate cabinet to my Tandy 1000. Although the Tandy already has two floppy drives and memory expansion board, I’d like the 20 Meg in a separate cabinet to serve as an expansion bay so I can add full-length boards at my wallet’s convenience.

I have three questions:

1. Is there any reason why I would want to retain both floppies in addition to the hard disk, and if so, would this set-up confuse MS-DOS (version 2.11)?
2. Could the hard drive be mounted
in the expansion bay and transmit the data across the cable which connects the bay (I assume the bay would be connected through a board which occupies one expansion slot in the computer and runs a cable of connections from the expansion bus out of the back of the computer)?

3. Is there any relatively cheap way to build an expansion bay without a lot of technical know-how? I've worked on CPU-controlled cash registers, so I'm not a complete idiot, but I don't feel extremely comfortable in a digital hardware environment which doesn't have step-by-step instructions.

Paul K. Vallandigham
2828 Dorothy Pl.
Orange CA 92669

Editor's note:
MS-DOS 2.11 certainly supports at least four floppies in addition to one or more hard drives, so software isn't a problem. Hardware might be; however, You mention extending the PC bus to a second cabinet. Someone was displaying such an expansion at Comdex last Fall. Unfortunately I didn't get the name, and I haven't heard about the expansion since.

Getting signals cleanly from one cabinet to the other would be a problem. You'd need a short, heavy, shielded cable. After all, you have power as well as signals on the bus. Capacitance between wires would also be a problem. This is not a trivial project.

Users Group Wanted
The Long Beach, CA area has needed a CP/M Kaypro user's group for a long time. I'd really enjoy getting together with other users of any level of expertise for some technical talk and support. If you're a Kaypro user in the Long Beach area and want to share your technical knowledge, please contact me.

Skipp Miller
405 E. Pacific Coast Hwy.
Long Beach CA 90806
(213) 591-1173 Bus.
(213) 599-3663 Res.

Needs Reader Input
I'm looking for a speech synthesis device for a possible portable computer application and would appreciate leads on either a Kaypro 2000 unit that would fit where the optional Kaypro modem would go or a unit for one of the mini boards (Ampro LB, Ampro LB/186, MicroMint, Southern Pacific, etc.) that doesn't consume an I/O port (perhaps using a SCSI interface).

Can anyone help?
Meredith A. Cargill
URH 248 Sherman Hall
909 S. Fifth St.
Champaign IL 61820

Letter Of Thanks
If you can comprehend the gratitude of one who just saved a whole disk of articles copied from microfilm by reading through a microscope, you know how I feel about UNERA19. My overwhelming thanks.

Dorothy Hoard
110 Sherwood
Los Alamos NM 87544

Does this look familiar?
What if each change you made to your program was ready to test in seconds instead of minutes?

"The SLR tools will change the way you write code. I don't use anything else." Joe Wright
New PC Speedup With RAM Resident Soft Switch

By Larry Fogg

Every once in a while an article just blows me away. This is definitely one of those articles. Larry not only figured out how to make his system into a screamer without affecting the system clock (or the signal to the color board), but he also came up with a memory resident program which changes system speed anytime it sees a special keystroke.

The hardware mod was very interesting, but writing the software was an even more involved project. Stay tuned while Larry walks you through the whole process.

Dave and I were lounging around the office a while back talking about speed. The subject came up because we'd had several requests for a PC speedup which wouldn't send the machine's real time clock (RTC) into warp speed.

We even had one person call to say that the original speedup (see Micro C issue 29) had no effect on his machine. Seems he'd used the RTC to time the benchmarks. Since the system clock and the RTC were both running faster the benchmark times didn't appear to change.

We consulted The Book of Intel (Component Data Catalog) and noticed that the PC clock generator, an 8284, could be switched between the normal crystal input and an external clock input. Off I went to dig into my computer. What follows is the results of my digging.

8284

The 8284 system clock chip supplies three different clock signals. CLK is the clock signal used by the processor, OSC feeds the expansion bus, and PCLK goes to the keyboard and RTC.

Switching from the 14.31818MHz crystal input to a higher frequency external source is easily done by toggling between high (5V) and low (ground) on the 8284's select pin. Unfortunately, this changes the value of all three clock outputs when we wanted to change only CLK.

The answer is to piggyback another 8284 on the existing one. The top 8284 will get the OSC output from the bottom one and will have its own higher frequency crystal oscillator circuit.

Now if we disconnect the CLK output of the bottom 8284 and instead supply the main board with CLK from the top 8284, a select signal to the top 8284 will switch between fast and slow CLK rates. The bottom 8284 is still providing OSC and PCLK to the system, so both the expansion bus and the RTC will stay happy.

Software/Hardware Interface

It would be good fun to control the clock speed with a software switch. No problem. There happens to be an unused bit in port B of the PIO. IBM says in their Technical Reference that it's bit 2, but a careful look at the schematic shows that bit 3 is really the one we want.

A wire from pin 21 (bit 3) of the PIO to the select pin of the top 8284 would finish the hardware end of the mod were it not for the fact that bit 3 is set high on power up. A high input to the top 8284 select will choose the external frequency which, in this case, is slow.

If there's one thing I can't stand it's watching a computer count through a memory check. So I added an inverter in the select line to finish things off. Now my box boots with blistering abandon.

On To The Nuts And Bolts

Take the main board out of your PC. You'll need to replace the 8088 with an
8MHz V-20 to handle the fast speed. Do it now, as the chip is much easier to install while the board is out.

The RAM chips are also sensitive to speed. You’re all set if the RAMs are 150 microsecond parts or faster. If they are slower, replace them with 150us or 120us.

Remove the 8284. Bend out pins 5 (READY) and 8 (CLK) since these signals will be provided by the top 8284. Prepare the top 8284 by doing the following. Bend out and cut the narrow portion off of pins 2, 5, 8, 12, 13, 14, 16, and 17. Sit the top 8284 on the bottom one and solder the remaining unbent pins to their downstairs neighbors. Check that none of the bent out pins is shorting out.

Solder a jumper wire from pin 12 of the bottom 8284 to pin 14 of the top 8284. This sends OSC from the bottom 8284 to the external input of the top 8284. Keep the jumper short, since it’s carrying high frequency and could pick up “I Love Lucy” if it starts to look too much like an antenna. Wire-wrap wire works well for all jumpers.

New Crystal Oscillator

Now comes the creative part. The top 8284 needs a crystal oscillator circuit to provide the higher frequency. I used the quick and dirty method (see Figure 1) and soldered directly to the pins of the top 8284. A small board would be much more aesthetic, but beware of long wires and high frequencies.

The fastest crystal I’ve been able to use is 22.1184MHz which gives a CLK (CLK = OSC/3) signal of 7.37MHz. With a 24MHz crystal the ROM signs on, but the drives go to lunch. Possibly the 5MHz DMA controller is failing.

Plug the monstrosity back in. Run a wire from pin 5 of the top 8284 around to the bottom of the board. Solder it to pad 5 of the 8284 socket to provide READY to the system. Put in a similar jumper from pin 8 of the top 8284 to pad 8 of the 8284 socket. This is the CLK signal.

Select Circuit

We’re almost there. The last step is to provide the select signal to the top 8284. I used a 74504 to invert the signal since we have buckets of them in the office. Chop off all of its legs except 1, 2, 7, and 14. Pay no attention to those tiny screams.

Bend out pins 1 and 2 and look for a convenient place to piggy back the chip. Any 14 pin DIP will do. Solder pins 7 and 14 to the bottom chip. Then wire a jumper from pin 21 of the PIO (it will be labeled 8255) to pin 1 of the 74504. Finally, jumper from pin 2 of the 74504 to pin 13 of the top 8284.

Reinstall the main board and fire it up. If all is well the next step is to program the PIO as a speed switch.

Support Software

SPEEDSET.ASM (see Figure 2) does three things. First it takes control of the keyboard interrupt. Then it carves out a portion of memory and makes itself resident. Finally it looks at all keystrokes and programs the PIO if it sees either the character that means “speed up” or the one that means “slow down.”

Let’s Corrupt An Interrupt

To do so we need to know how they work. The PC devotes the first section of its memory to a table of four-byte vectors. Each of these vectors points to the code for one of the interrupts.

(SPEEDSET.ASM on next page)
When a program calls an interrupt, INT 9 for example, a jump to the address stored in location 0000:4*9 occurs. This address is the beginning of the code for interrupt 9, which is then executed. To take control of an interrupt, write the address of our replacement code to the vector table in place of the normal vector.

In this case we want the keyboard interrupt (INT 9) to function normally after each keystroke is inspected. So one of the first things we do in the setup procedure is save the original vector.

DOS function 35h returns the vector which is then stored in the variable old_int. Notice that we guard against installing the program more than once. This is easy because the interrupt vector points to a location in the ROM if no one has altered it.

The beginning of the ROM has a segment address of 0f000h, so if function 35h returns that value, the program has not been installed.

Now install a new vector which points to the start of our resident code using DOS function 25h. Any further calls to INT 9 will be redirected to the new code. The last two instructions in setup protect the resident code from being overwritten. Interrupt 27h does all the work and only has to know the amount of memory to protect.

Scan Codes Vs. ASCII

Once we have control of the keyboard interrupt what do we do?

Each key of the PC keyboard generates a unique scan code rather than ASCII. There is also a byte in memory called kb_flag which contains the status of the various shift keys. Between the scan code and kb_flag any keystroke can be identified.

New_int looks for Alt F9 and Alt F10 as the speed switches. It needs to know first if the Alt key has been pressed and then whether the scan code for either F9 or F10 is in the kb_data port.

If neither speed key is found, control is passed on to old_int. Otherwise we jump to one of the two parts of the program which actually does something.

```assembly
section .data
kb_data equ 60h ; PIO port A - contains scan code
kb_ctl equ 61h ; PIO port B - contains unused bit and keyboard acknowledge bit
code segment ; everything goes in code segment
org 100h
assume cs:code

DOS_entry label far
jmp setup

old_int dd ? ; address of interrupt 9 code in ROM
new_int proc far ; beginning of our interrupt handler
sti
push ax
push bx
push dx
mov ah,2 ; tell INT 16 to return kb_flag
int 16h ; get kb_flag
and al,00001000b ; mask bits except alt status
cmp al,00001000b ; check alt key status
jne short no_act ; don't act if alt not pressed
in al,kb_data ; get scan code
cmp al,f9 ; is it the f9 key?
je short faster ; speed up if it is
cmp al,f10 ; is it the f10 key?
je short slower ; slow down if it is

short no_act: pop dx ; restore registers
pop bx
pop ax
jmp old_int ; let ROM code do it's thing

short faster: in al,kb_ctl ; get keyboard status
or al,11110111b ; set keyboard acknowledge bit
out kb_ctl,al ; write it back to port
pop dx
pop bx
pop ax
jmp old_int ; jump to ROM code to finish

short slower: in al,kb_ctl ; get value from PIO port B
or al,00000000b ; reset unused bit
out kb_ctl,al ; write it back to port B
jmp short done ; finish

short done: in al,kb_ctl ; get value from PIO port B
out kb_ctl,al ; set unused bit
jmp short done ; finish

new_int endp

end of our interrupt
```
Ever Wondered What Makes TURBOPASCAL Tick?

Source Code Generators by C. C. Software can give you the answer.

"The darndest thing I ever did see..."
"...if you're at all interested in what's going on in your system, it's worth it."

The SCG-TP program produces fully commented and labeled source code for your TURBO-Pascal system. To modify, just edit and assemble. Version 3.00A (280) is $45. SCG's available for CP/M 2.2 ($45) and CP/M+ ($75). Please include $1.50 postage (in Calif add 6.5%).

C. C. Software, 1907 Alvarado Ave.
Walnut Creek, CA 94596 (415)939-8153

CP/M is a registered trademark of Digital Research, Inc.
TURBO Pascal is a trademark of Borland International

The code at labels "faster:" and "slower:" toggles bit 3 of PIO port B. This in turn toggles pin 21 of the PIO and selects between the two CLK rates. Finally new_int sets the keyboard acknowledge bit of PIO port B so it will be ready for the next keystroke.

EXE To COM Conversion
SPEEDSET.ASM was written to be assembled as an EXE file and then run through EXE2BIN to create a COM file. Those of you who feel adventurous can use the assembler file as a shell to install your own memory resident programs. Replace the speed change code with your own and dive in. There are a few rules to keep EXE2BIN from choking on your EXE file.

Don't include separate stack or data segments. All segment registers should be assumed to refer to the code segment and any variables should be in the code segment.

Begin the program with "ORG 100h." Label the first instruction and end the program with "END label_name." And don't worry about the stack segment error from LINK. It's just confused.

Non-Resident Speed Switch
If you really don't care about a memory resident speed switch use DEBUG to create two small programs using the four lines of code at "faster:" and "slower:". Just replace the jump instruction at the end of each code segment with an INT 20h to exit to DOS.

Conclusion
I've had this mod running on my PC for a couple of months with no problems. The software works well. Most programs don't even care if the speed is changed in the middle of execution. I did manage to make Turbo Pascal gag during a compilation, but why anyone would want to change speeds then escapes me.

Good luck. When you're done, if the digital gods are smiling, you'll have a 7.37MHz PC that can actually tell time.

MICRO CORNUCOPIA, #31, Aug-Sept 1986 9
**SPECIALS**

**Bahno Wire Cutter “Swedish”** .......................... 6.95

**CPU/SUPPORT**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>MC68000-8 CPU</td>
<td>$12.50</td>
</tr>
<tr>
<td>CB0186-3</td>
<td>27.00</td>
</tr>
<tr>
<td>CB0186-6</td>
<td>9.00</td>
</tr>
<tr>
<td>280 C0</td>
<td>1.00</td>
</tr>
<tr>
<td>280A CPU</td>
<td>2.50</td>
</tr>
<tr>
<td>280A PIO</td>
<td>2.00</td>
</tr>
<tr>
<td>280A S10</td>
<td>5.00</td>
</tr>
<tr>
<td>280A DMA</td>
<td>10.00</td>
</tr>
<tr>
<td>8088</td>
<td>10.00</td>
</tr>
<tr>
<td>8089-3</td>
<td>29.00</td>
</tr>
<tr>
<td>D8047 CPU/EPROM 12 MHz</td>
<td>20.00</td>
</tr>
<tr>
<td>D8284A</td>
<td>3.00</td>
</tr>
<tr>
<td>D8288</td>
<td>9.00</td>
</tr>
<tr>
<td>4164-15</td>
<td>1.10</td>
</tr>
<tr>
<td>4125-15</td>
<td>2.50</td>
</tr>
<tr>
<td>4125-12</td>
<td>2.75</td>
</tr>
<tr>
<td>6116 2Kx8 Static Ram</td>
<td>1.50</td>
</tr>
</tbody>
</table>

**MODEM**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMS 99532 (Modem)</td>
<td>$12.00</td>
</tr>
<tr>
<td>TMS 99531 (Dialer)</td>
<td>8.00</td>
</tr>
</tbody>
</table>

**FLOPPY CONTROLLER**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>WD 1791</td>
<td>$9.00</td>
</tr>
<tr>
<td>WD 1793</td>
<td>5.00</td>
</tr>
<tr>
<td>WD 1797</td>
<td>7.00</td>
</tr>
<tr>
<td>WD 2797</td>
<td>9.50</td>
</tr>
</tbody>
</table>

**BAUD GENERATOR**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>WD 8116</td>
<td>$4.50</td>
</tr>
</tbody>
</table>

**MISCELLANEOUS**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>UC3524 Sw. Reg.</td>
<td>$8.50</td>
</tr>
<tr>
<td>MM58167A Clock Chip</td>
<td>5.50</td>
</tr>
<tr>
<td>CA301 Op Amp</td>
<td>3.50</td>
</tr>
<tr>
<td>CA083 Op Amp</td>
<td>4.50</td>
</tr>
<tr>
<td>2N3055</td>
<td>7.00</td>
</tr>
<tr>
<td>2N6852</td>
<td>1.00</td>
</tr>
<tr>
<td>LM323K Volt. Reg. + 5V/3A</td>
<td>4.50</td>
</tr>
<tr>
<td>LM78H12 Volt. Reg. + 12V/5A</td>
<td>5.00</td>
</tr>
</tbody>
</table>

**VIDEO CONTROLLER**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>6545</td>
<td>$7.00</td>
</tr>
<tr>
<td>6545A-1</td>
<td>8.00</td>
</tr>
<tr>
<td>6845</td>
<td>6.00</td>
</tr>
</tbody>
</table>

**EPROMS**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>2716</td>
<td>$3.50</td>
</tr>
<tr>
<td>2732</td>
<td>3.50</td>
</tr>
<tr>
<td>2764</td>
<td>2.25</td>
</tr>
<tr>
<td>2764-25</td>
<td>3.50</td>
</tr>
</tbody>
</table>

**74LS SERIES**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>HC00</td>
<td>$3.50</td>
</tr>
<tr>
<td>HCU04</td>
<td>4.00</td>
</tr>
<tr>
<td>LS 125</td>
<td>4.00</td>
</tr>
<tr>
<td>LS 373</td>
<td>6.00</td>
</tr>
<tr>
<td>LS 375</td>
<td>6.00</td>
</tr>
<tr>
<td>4073</td>
<td>3.50</td>
</tr>
<tr>
<td>7406</td>
<td>4.00</td>
</tr>
</tbody>
</table>

**PRINTERS**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Centronics 703</td>
<td>$395.00</td>
</tr>
<tr>
<td>Centronics 588 - As Is</td>
<td>125.00</td>
</tr>
<tr>
<td>NEC2000 w/Single Sheet Feed</td>
<td>$1200.00</td>
</tr>
</tbody>
</table>

**SOLAR CELLS**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>2½&quot; x 5/16&quot;, .5V/.5A</td>
<td>$2.25</td>
</tr>
</tbody>
</table>

**IBM PC/XT COMPATIBLES**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monochrome Card</td>
<td>$67.50</td>
</tr>
<tr>
<td>Cherry Keyboard</td>
<td>$37.50</td>
</tr>
<tr>
<td>Goldstar Model 4333</td>
<td>199.00</td>
</tr>
<tr>
<td>Goldstar Model MBM2054</td>
<td>Hercules compatible</td>
</tr>
</tbody>
</table>

**POWER SUPPLIES**

**WALL PLUG-IN TYPE**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>6VAC/.95A</td>
<td>$1.35</td>
</tr>
</tbody>
</table>

**LINEAR**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>14KV DC/DC TV Power Supply</td>
<td>4.50</td>
</tr>
<tr>
<td>+ 5V/1A</td>
<td>6.50</td>
</tr>
<tr>
<td>+ 5V/2A</td>
<td>12V/1A</td>
</tr>
<tr>
<td>+ 24V/2.2A</td>
<td>29.00</td>
</tr>
</tbody>
</table>

**SWITCHERS**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>5V/9.5A</td>
<td>12V/3.8A</td>
</tr>
<tr>
<td>5V/3A</td>
<td>12V/2A</td>
</tr>
<tr>
<td>5V/10A</td>
<td>25.00</td>
</tr>
<tr>
<td>5V/75A</td>
<td>12V/8A</td>
</tr>
<tr>
<td>5V/20A</td>
<td>195.00</td>
</tr>
</tbody>
</table>

**FLOPPY DISK DRIVES**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Okidata DSDD</td>
<td>$39.00</td>
</tr>
<tr>
<td>Mitsubishi M4851 DSDD 60 Tr.</td>
<td>$139.00</td>
</tr>
<tr>
<td>Qume Trak 142 1/4&quot; DSD</td>
<td>$99.00</td>
</tr>
<tr>
<td>Tandon TM 889-1 8&quot; SSD</td>
<td>149.00</td>
</tr>
<tr>
<td>Shugart 475 DS Quad 1.2Mb</td>
<td>159.00</td>
</tr>
<tr>
<td>Shugart SA860 DS Quad - As is</td>
<td>$36.00</td>
</tr>
<tr>
<td>Remex RDF4000</td>
<td>199.00</td>
</tr>
<tr>
<td>Persci 277 2x8&quot;</td>
<td>399.00</td>
</tr>
</tbody>
</table>

**TERMINALS**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>IntErTube II Smart (B&amp;W)</td>
<td>$195.00</td>
</tr>
<tr>
<td>IntColor 3602 Smart (Color)</td>
<td>329.00</td>
</tr>
</tbody>
</table>

**MISCELLANEOUS**

<table>
<thead>
<tr>
<th>Model</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Headset/Boom Microphone</td>
<td>$3.95</td>
</tr>
<tr>
<td>Nicad Pack 12V/.5AH</td>
<td>5.00</td>
</tr>
<tr>
<td>Joystick 4 Switches</td>
<td>9.00</td>
</tr>
<tr>
<td>NEC2000 w/Single Sheet</td>
<td>295.00</td>
</tr>
<tr>
<td>Yamaha 789</td>
<td>9.50</td>
</tr>
<tr>
<td>Elgar 400W Unint. Power</td>
<td>375.00</td>
</tr>
<tr>
<td>Elgin 10 Ft. Line Cord</td>
<td>1.65</td>
</tr>
</tbody>
</table>
This is the Modula-2 compiler everybody's been waiting for...

Especially Turbo Pascal users!

Logitech Modula-2/86

Logitech Modula-2 is at its absolute best. It's a fully integrated development environment that takes into account what you need as a programmer. Without leaving the Editor, you can call the compiler, linker and utilities.

With Logitech's Modula-2, you'll have the ability to edit several files at once, comparing, window to window, various code modules. You can even move from window to window compiling, linking, debugging and running.

Compiler has the kind of power and room to breathe that you really need in today's complex applications. It is as easy to use as Turbo Pascal, without your programs being limited to 64K of code.

Your command will be the libraries of modules that make Modula-2 a programmer's dream. It has essentially the same structure as Pascal with the major addition of a library organization of code modules that allow you to put together programs on a solid, block-by-block, foundation of proven code.

Whether you're working with a module of your own making, or one of the many in our library, you'll find the system by which each module is identified, described and stored an organizational masterpiece. And that's at the heart of Modula-2.

Underneath the sophisticated system is a Modula-2 compiler that is the result of years of development and proven use in industry. We run on the Vax®, and we run on the IBM PC. And the code is portable—from one to the other.

Best of all... you can have it right now!

Logitech Modula-2/86 Complete with Editor, Run Time System, Linker, Cursor-positioning debugger, 8087 Software Emulation, BCD module, Logitech's extended library, Utility to generate standard .EXE files, Turbo Pascal (and standard Pascal, too) to Modula-2 translator (included without charge until 8/1/86), and much, much more!

Logitech Modula-2/86 with 8087 support Even if you haven't yet gotten an 8087 coprocessor, you can still use this version.

Logitech Modula-2/86 Plus For machines with 512K or more. Takes advantage of the larger memory to increase compilation speed by 50%! Supports 80186 and 80286 as well as 8086 and 8088. Includes 8087 and 80287 support, too.

Window Package Now you can build true windows into your Modula-2/86 code with ease. Very powerful and very full, yet only 15K in size. Features virtual screens, color support, overlapping windows and a variety of borders.

Run Time Debugger (source level) Much more powerful than just a symbolic RTD. Display source code, data, procedure call chain and raw memory. Set break points, assign values to variables, pinpoint and identify bugs in your source. The ultimate professional's tool!

Utilities Package Features a post-mortem debugger for static debugging. If a program you've written crashes at run time, the situation is frozen, and you can pinpoint, in source, the cause of the error and the data at that moment. Also includes a disassembler, a cross reference utility and a "version" utility that allows conditional compilation.

Make Utility Automatically selects modules affected by code changes for quick and minimal re-compilation and relinking. Even figures out dependencies for you.

Library Sources Source code for our major library modules is now available—for customization or exemplification.

ROM Package If you need to produce runnable code, call our $60 number for further information on this package.

To place an order call our special toll free number

800-231-7717 in California
800-552-8885 Special offer until 8/1/86! includes

Free! $49.95 value Turbo Pascal translator!
Now, you can take your library with you!

I'd like to take the next logical step in programming.
Please send my copy of Logitech Modula-2/86 to the following address:

Card Number Expiration Date

Name ___________________________ Signature ___________________________
Address __________________________ City __________________________
State ______ Zip ______ Phone ______

Here's the configuration I'd like: And include the indicated items:

□ Logitech Modula-2/86 $89
□ Logitech Modula-2/86 Plus $129
□ Logitech Modula-2/86 with 8087 support $129
□ Make Utility $29
□ Library Sources $19
□ Window Package $49
□ Utilities Package $49
□ Run Time Debugger $69

Please add $6.50 for shipping and handling.

Total enclosed $_____

(Tax residents, please add applicable sales tax)

Logitech

Logitech, Inc.
805 Veterans Boulevard
Redwood City, California 94063
Telephone (415) 365-9852

For European pricing, please contact:

LOGITECH SA
Box 32, CH-1143 Apples, Switzerland
Telephone 41 (21) 774-545

Turbo Pascal is a registered trademark of Borland International.
Practical Programming In Modula-2:

Controlling The Video On PCs

You may be wondering where we dug up a FORTRAN user who's still interested in programming (maybe it's because he's an anthropologist), but Jim actually turned himself in.

He's excited about Modula-2, and so am I. I too learned FORTRAN in school, and I too know its frustrations only too well. Read on as Jim does some interesting things using a very solid (and now inexpensive) implementation of this great new language.

When I asked a friend to teach me how to program a computer back in the mid '70s, he tried to teach me FORTRAN.

When I entered graduate school in the early '80s, I was again confronted with FORTRAN, but this time I had an incentive I lacked earlier: enormous amounts of data to process. So I learned.

When I bought my first micro, I bought a FORTRAN compiler for it. And when I landed my first sizeable programming contract, I wrote the code in FORTRAN. So why did I decide to move beyond FORTRAN?

Life After FORTRAN

First, the FORTRAN implementations I had didn't allow separate compilation of pieces of a large program. Routine libraries were possible, but clumsy in comparison to more modern languages.

Low-level operations were limited at best, graphics difficult. And while the formatted WRITE statement is fine for statistical programming, creating output for a "user-friendly" application is tedious. Recursion can be faked but results in truly opaque code. So it was time to go shopping for a new language.

I wanted these features — recursion, user-defined complex variable types, separate compilation of components of larger programs, low-level control of

Figure 1 - DEFINITION MODULE

DEFINITION MODULE MonoPage;
FROM FileSystem IMPORT File;
EXPORT QUALIFIED (* variable types: *)
Window,
Page,
Pagepointer,
(* constants: *)
page0,
(* procedures: *)
MakePage,
UsePage,
SwapPage,
Store2Page,
RemovePage,
MergePage,
WriteWindowChar,
WriteWindowString,
ChangeWindowAttribute,
ClearWindow,
WritePageChar,
WritePageString,
ChangeCharAttribute,
ClearPage,
DumpPage,
LoadPage;

CONST page0 = 0B000H:0H; (* this is the monochrome version! *)

TYPE Window = ARRAY [ 0 .. 3 ] OF CARDINAL;
Charrec = RECORD
code,
attr : CHAR;
END;
Page = ARRAY [ 1 .. 25 ],
[ 1 .. 80 ] OF Charrec;
Pagepointer = POINTER TO Page;

PROCEDURE MakePage ( VAR page : Pagepointer );
PROCEDURE UsePage ( page : Pagepointer );
PROCEDURE SwapPage ( page : Pagepointer );
PROCEDURE Store2Page ( VAR page : Pagepointer );
PROCEDURE RemovePage ( VAR page : Pagepointer );
PROCEDURE MergePage ( basepage,
                       overpage : Pagepointer;
                       VAR rsltpage : Pagepointer;
                       resolution : CARDINAL );
PROCEDURE WriteWindowChar ( inpchar : CHAR;
                             page : Pagepointer;
                             window : Window;
                             row,
                             col : CARDINAL );
PROCEDURE WriteWindowString ( string : ARRAY OF CHAR;
                                page : Pagepointer;
                                window : Window );
PROCEDURE ChangeWindowAttribute ( attrib : CHAR;
                                   page : Pagepointer;
                                   window : Window );
PROCEDURE ClearWindow ( page : Pagepointer;
                       window : Window );
PROCEDURE WritePageChar ( inpchar,
                         attrib : CHAR;
                         page : Pagepointer;
                         row,
                         col : CARDINAL );
PROCEDURE WritePageString ( string : ARRAY OF CHAR;
                          attrib : CHAR;
                          page : Pagepointer;
                          row,
                          col : CARDINAL );
PROCEDURE ChangeCharAttribute ( attrib : CHAR;
                        page : Pagepointer;
                        row,
                        col : CARDINAL );
PROCEDURE ClearPage ( page : Pagepointer );
PROCEDURE DumpPage ( page : Pagepointer );
PROCEDURE LoadPage ( fname : ARRAY OF CHAR;
                    page : Pagepointer );
END MonoPage.

graphics, and support for my interests in laboratory instrumentation and control.
I had to have a good general-purpose language, and the code had to be readable and easily maintained.
In order to minimize the confusion inherent in large projects, I wanted strong type-checking of variables, but I also wanted enough control to corrupt the type-checking if I needed to.
All this added up, it seemed to me, to Modula-2. When Logitech lowered the price of their compiler to less than one hundred dollars, I ordered.

From WRITE To WRITELN
After playing with my new compiler for a while, I decided to translate a large FORTRAN program into Modula-2. There were 26 pages and a lot of lines starting with "WRITE". Unfortunately, the Modula-2 alternative, "WRITELN", convinced me to put the project on hold.
Modula has a number of procedures for handling output, and its modular structure allows for groups of procedures called (strangely enough) "modules" to be collected and compiled. These procedures can then be used by a program simply by explicitly importing them.
All input and output require importation of procedures, usually from the library modules InOut and Terminal. These procedures are type-specific: WriteCard, for example, writes only cardinal numbers.
None of these procedures issues a carriage return/line feed, so another procedure, WriteLn, must be used if you want to add one. With all the screen writing I had to do with this program, I needed more control.
What I wanted was a set of procedures that would allow me to write to specified locations on the screen, with full control of screen display attributes (bold, reverse, underlined, blinking).
I wanted to get the job done without ANSI.SYS or direct BIOS calls. And I wanted the ability to overwrite material on the screen and then restore it a la windows. Finally, I wanted to store entire screens in memory and then recall them when needed.

(continued next page)
(continued from page 13)

The Answer In Theory

PCs have essentially two types of video displays: monochrome and color. The former produces dense, highly readable text characters and an assortment of block graphics characters, while the latter offers less readable text but greater flexibility, with the possibility of 40 column text displays and dot graphics.

Most of the people I know have monochrome, since they primarily do word processing on their computers. So I write assuming mono.

The memory that represents your screen display lives on the video card. High resolution graphics are very memory expensive, so color and mono graphics cards contain lots of memory.

Since a display consisting of standard characters eats up only about 4K of memory when in “text mode,” a bunch of bytes are left over. MS/PC-DOS allows this to be carved up into “pages” which can be selected for display with a BIOS command.

Since monochrome text always takes up the same 4K chunk of memory, that’s all mono boards have on them. No pages are possible. PCs, on the other hand, usually have lots of memory (on the main board) for transient programs. The goal, then, is to use some of that memory to emulate pages, or create “virtual pages” if you will, in user RAM.

And In Modula

I learned from a couple of sources how to treat my screen as a big variable —

Define a record with two character elements, the first representing the character to display, the second its attribute. Set up a two-dimensional array using this record, resulting in a complex variable that represents the display.

Modula allows a memory location to be specified for a variable, and in this case we want to locate the screen variable on the monochrome video RAM. So, saying something like “screen [3,10].character := ‘P’” would put a capital P on the screen in the third row, tenth column.

Pointer variables are just what we need since they’re defined as type “pointer to” another type and contain an address.

The “virtual pages” can be constructed (using the same screen-type variable) somewhere in memory by importing the procedures ALLOCATE and DEALLOCATE from the standard library module Storage, and using the call NEW (pointer) to reserve an appropriately sized chunk of RAM, with its address returned in pointer.

The screen referred to by the pointer can be directly manipulated by “dereferencing” the pointer by putting the hat symbol ‘^’ directly after the variable name, thus: pointer.

Modula won’t allow the passing of complex variables as parameters in function calls, but will allow the passing of pointers. This lets you build a library of procedures that manipulates screens.

I call this library module “MonoPage.” (See Figures 1 and 2 for the complete Definition and Implementation modules.) It features procedures that create pages, move them into the current display, swap them with the current display, open windows on any of the pages, change attributes and clear pages and windows, write individual characters and strings, etc.

Then other goodies came to mind. Help screens are important, and the screen dump feature of the PC lets the user get a hard-copy of instructions, but the screen display attributes are lost (can you imagine the printer trying to figure out on its own what to do with a blinking character?). I also couldn’t assume the user would have GRAPHICS.COM and an Epson-compatible printer.

I wanted a page dump routine that would boldface the bold, underscore the underlined, and italicize the reverse (and ignore blinking!). So I added DumpPage.

If you know where and how big a variable is, you can read or write its contents to and from a disk file. An elaborate screen can be created and saved, and then read back in by the application with no write-type coding in that program at all. This led to LoadPage.

I like using the block graphics characters for bar graphs. It would be neat to overlay two pages of graphs, but something needs to be done to resolve what happens when they overlap. All I’ve done so far is allow for direct, simple overlay in procedure MergePage.

Windows are handy little things, but what happens if you want to print an 85-character string to a window only 20 characters wide? To avoid unnatural breaks in words at the window’s right edge, I wrote a WordWrap procedure as a private procedure of MonoPage (it can’t be exported).

MonoPage isn’t perfect, but it answers most of my needs for screen output. Most of the ideas behind it are simple and straightforward, if not downright obvious, but it would have been very difficult to do in FORTRAN. That a beginner in Modula programming could do so with relative ease speaks well for the power and simplicity of this language.

Further Thoughts Of Modula

Modula’s library modules are implemented in two parts: a definition module, defining what procedures, constants, and types are available for export, and an implementation module that contains the working routines. Both need to be compiled, and the definition module must be compiled before the implementation module.

The definition module is all a user has to see to be able to use its procedures, with many details of the implementation, like the WordWrap procedure in my MonoPage, going largely unsuspected by the user.

Here, I think, is a fundamental difference between FORTRAN and Modula. Reusable routines can be written and edited in FORTRAN, or assembly language can be called from a program. But these measures feel like appendages and are always recognizable as additions.

A well-conceived Modula module becomes indistinguishable from the other modules supplied with the language, and the sense is that the language itself is being extended. The programming environment therefore becomes “co-evolutionary."

Modula-2 could be hard to outgrow.

---

14  MICRO CORNUCOPIA, #31, Aug-Sept 1986
IMPLEMENTATION MODULE MonoPage;

FROM SYSTEM IMPORT TSIZE;

FROM Terminal IMPORT Read;

FROM Storage IMPORT ALLOCATE, DEALLOCATE;

FROM FileSystem IMPORT File, ReadNBytes, WriteChar, Lookup, SetRead, Close;

= 040C;
= 015C;
= 011C;
= 170C;
= 007C;
= 001C;
= 160C;

CONST blank = 040C;
normal = 015C;
undrln = 011C;
revers = 170C;
normal = 007C;
undrln = 001C;
revers = 160C;
blink = 207C; (* "blink" is normal video, normal intensity *)
bLink = 217C; (* "bLink" is normal video, increased intensity *)
flash = 360C; (* "flash" is reverse video, normal intensity *)
flash = 370C; (* "flash" is reverse video, increased intensity *)
ulc = 311C; (* upper-left corner char *)
urc = 273C; (* upper-right corner char *)
llc = 310C; (* lower-left corner char *)
lrc = 274C; (* lower-right corner char *)
hl = 315C; (* horizontal line char *)
vl = 272C; (* vertical line char *)

VAR screen [ page0 ]; Page;
query : CHAR;

PROCEDURE DefineAttrib ( attrib : CHAR ) : CHAR;

VAR trnschar : CHAR;
BEGIN

CASE attrib OF
'N' : trnschar := normal |
'u' : trnschar := undrln |
'r' : trnschar := revers |
'N' : trnschar := normal |
'u' : trnschar := undrln |
'r' : trnschar := rEvers |
'b' : trnschar := blink |
'B' : trnschar := blink |
'F' : trnschar := flash |
'f' : trnschar := flash |
ELSE trnschar := blank;
END;

PROCEDURE DefineWindow ( page : Pagepointer; window : Window );

VAR i, top, bottom, left, right : CARDINAL;
BEGIN

top := window [ 0 ];
bottom := window [ 1 ];
left := window [ 2 ];
right := window [ 3 ];

WritePageChar ( ulc, 'N', page, top, left );
WritePageChar ( urc, 'N', page, top, right );
WritePageChar ( llc, 'N', page, bottom, left );
WritePageChar ( lrc, 'N', page, bottom, right );

FOR i := ( left + 1 ) TO ( right - 1 ) DO
  WritePageChar ( hl, 'N', page, top, i );
  WritePageChar ( hl, 'N', page, bottom, i );
END;

FOR i := ( top + 1 ) TO ( bottom - 1 ) DO
  WritePageChar ( vl, 'N', page, i, left );
  WritePageChar ( vl, 'N', page, i, right );
END;

END DefineWindow;

PROCEDURE Scroll ( strow, edrow, stcol, edcol ; page : Pagepointer );

VAR rp, rrp, cp, i, j, delay : CARDINAL;
BEGIN

delay := edcol - stcol;
delay := delay * delay;
FOR i := 1 TO 100 DO
  FOR j := 1 TO delay DO
    rp := strow TO edrow DO
      rrp := rp + 1;
    FOR cp := stcol TO edcol DO
      page [ rp, cp ] := page [ rrp, cp ];
  END;
END;

(code continued next page)
edrow := edrow + 1;
FOR cp := startcol TO edcol DO
  page^ [ edrow, cp ].code := ' ';
END;
END Scroll;

PROCEDURE WordWrap ( startrow,
  edrow,
  startcol,
  edcol,
  row : CARDINAL;
  VAR col : CARDINAL;
  page : Pagepointer);

VAR hold,
  step : CARDINAL;

BEGIN
  IF row = endrow THEN
    Scroll( startrow, endrow, startcol,
      edcol, page);
    page := page^ [ edrow ];
    END;
  endcol;
  hold := endcol;
  REPEAT
    DEC ( hold );
    UNTIL page^ [ row, hold ].code = ' ';
  hold := hold + 1;
  FOR step := hold TO endcol DO
    page^ [ row + 1, col ] := page^ [ row, step ];
    page^ [ row, step ].code := ' ';
    INC ( col );
  END;
END WordWrap;

PROCEDURE MakePage ( VAR page : Pagepointer );

BEGIN
  NEW ( page );
  ClearPage ( page );
  END MakePage;

PROCEDURE SwapPage ( page : Pagepointer );

VAR holdpage : Pagepointer;

BEGIN
  NEW ( holdpage );
  holdpage^ := screen;
  screen := page^;
  page^ := holdpage^;
  DISPOSE ( holdpage );
  END SwapPage;

PROCEDURE UsePage ( page : Pagepointer );

BEGIN
  screen := page^;
  END UsePage;

PROCEDURE RemovePage ( VAR page : Pagepointer );

BEGIN
  DISPOSE ( page );
  END RemovePage;

PROCEDURE MergePage ( basepage,
  overpage : Pagepointer;
  VAR rsltpage : Pagepointer;
  resolution : CARDINAL );

VAR i,
  j : CARDINAL;

(* THIS PROCEDURE HAS ONLY ONE "RESOLUTION"
CURRENTLY IMPLEMENTED: SIMPLE, DIRECT OVERLAY
OF THE BASE PAGE WITH THE OVER PAGE *)

BEGIN
  IF rsltpage = NIL THEN MakePage ( rsltpage );
  END;
  FOR i := 1 TO 25 DO
    FOR j := 1 TO 80 DO
      IF overpage^ [ i, j ].code # blank THEN
        rsltpage^ [ i, j ] := overpage^ [ i, j ]
      ELSE
        rsltpage^ [ i, j ] := basepage^ [ i, j ]
      END;
    END;
  END;
END MergePage;

PROCEDURE WritePageString (string: ARRAY OF CHAR;
  attrib: CHAR;
  VAR i,
  page : Pagepointer;
  row,
  col : CARDINAL );

BEGIN
  attrib := DefineAttrib ( attrib );
  i := 0;
  WHILE i <= HIGH ( string ) DO
    WritePageChar ( string [ i ], attrib, page,
      row, col );
    IF col = 80 THEN
      IF string [ i ] # blank THEN
        WordWrap (1,25,1,80,row,col,page);
      ELSE
        col := 1;
      END;
    END;
  END;
END WritePageString;
INC ( row );
IF row >= 25 THEN
    Scroll ( 1, 25, 1, 80, page );
    row := 25;
END;
ELSE
    INC ( col );
END;
INC ( i );
END;
END WritePageString;

PROCEDURE ClearPage ( page : Pagepointer );
VAR i,
    j : CARDINAL;
BEGIN
    FOR i := 1 TO 25 DO
        FOR j := 1 TO 80 DO
            WritePageChar ( blank, 'n', page, i, j );
        END;
    END;
END ClearPage;

PROCEDURE WriteWindowString (string:ARRAY OF CHAR;
    page :Pagepointer;
    window: Window );
VAR i,
    j,
    k,
    loop,
    row,
    col,
    endrow,
    endcol : CARDINAL;
BEGIN
    DefineWindow ( page, window );
    row := window [ 0 ] + 1;
    endrow := window [ 1 ] - 1;
    col := window [ 2 ] + 1;
    endcol := window [ 3 ] - 1;
    i := 0;
    j := row;
    k := col;
    WHILE i <= HIGH ( string ) DO
        WritePageChar ( string [ i ], blank, page, j, k );
        IF k = endcol THEN
            IF string [ i ] # blank THEN
                WordWrap ( row, endrow, col, endcol, j, k, page );
                INC ( j );
            ELSE
                k := col;
                INC ( j );
            END;
        END;
        j := endrow;
        IF j >= endrow THEN
            j := endrow;
            END;
        ELSE
            INC ( j );
            END;
        INC ( i );
    END;
END WriteWindowString;

PROCEDURE ClearWindow (page : Pagepointer;
    window : Window );
VAR i,
    j : CARDINAL;
BEGIN
    FOR i := window [ 0 ] TO window [ 1 ] DO
            WritePageChar ( blank, 'n', page, i, j );
        END;
    END;
    DefineWindow ( page, window );
END ClearWindow;

PROCEDURE ChangeWindowAttribute (attrib : CHAR;
    page : Pagepointer;
    window : Window );
VAR i,
    j : CARDINAL;
BEGIN
    attrib := DefineAttrib ( attrib );
    FOR i := window [ 0 ] TO window [ 1 ] DO
            page [ i, j ].attr := attrib;
        END;
    END;
    DefineWindow ( page, window );
END ChangeWindowAttribute;
IMPLEMENTATION MODULE
(continued from page 17)

PROCEDURE WriteWindowChar ( inpchar:CHAR; 
page :Pagepointer; 
window :Window; 
row, 
col :CARDINAL );

BEGIN 
row := window \[ 0 \] + row;
col := window \[ 2 \] + col;
WritePageChar (inpchar,blank,page,row,col);
END WriteWindowChar;

PROCEDURE WritePageChar ( inpchar, 
attrib : CHAR; 
page : Pagepointer; 
row, 
col : CARDINAL );

BEGIN 
attrib := DefineAttrib ( attrib );
page^ \[ row, col \].code := inpchar;
IF attrib \# blank THEN 
page^ \[ row, col \].attr := attrib;
END;
END WritePageChar;

PROCEDURE ChangeCharAttribute (attrib:CHAR; 
page :Pagepointer; 
row, 
col :CARDINAL );

BEGIN 
attrib := DefineAttrib ( attrib ) ;
page^ \[ row, col \].attr := attrib;
END ChangeCharAttribute;

PROCEDURE DumpPage ( page : Pagepointer );

(* THIS DumpPage LACKS PRINTER CONTROL 
FEATURES DUE TO THE LACK OF A COMMON 
STANDARD FOR SUCH CODES. ALL GRAPHICS 
CHARACTERS ARE ALSO FILTERED OUT *)

VAR i, 
j;
outval : CARDINAL;
prindump : File;
outchar : CHAR;
BEGIN

Lookup ( prindump, 'PRN', FALSE );
SetWrite ( prindump );

FOR i := 1 TO 25 DO
  FOR j := 1 TO 80 DO
    outchar := pageA[ i, j ].code;
    outval := ORD ( outchar );
    IF ( outval > 32 ) & (outval < 128) THEN
      WriteChar ( prindump, outchar );
    ELSE
      WriteChar ( prindump, blank );
    END;
  END;
END;

Close ( prindump );

END DumpPage;

PROCEDURE LoadPage ( fname : ARRAY OF CHAR; page : Pagepointer );

VAR readfile : File;
  size, got : CARDINAL;
BEGIN
  size := TSIZE ( Page );
  got := 0;
  Lookup ( readfile, fname, FALSE );
  SetRead ( readfile );
  ReadNBytes ( readfile, page, size, got );
  Close ( readfile );
END LoadPage;

PROCEDURE Store2Page ( VAR page : Pagepointer );

BEGIN
  IF page = NIL THEN MakePage ( page ); END;
END Store2Page;

END MonoPage.

End of IMPLEMENTATION MODULE
dare to compare.

Super Turbo
V20 - 8mHz
$1299

This Super Turbo P.C. runs IBM software and CPM 8080 programs

- CPU - V20-8 8mHz Super Chip runs "IBM compatible software at 3 times the speed of the IBM-XT and CPM 8080 software.
- 8087 Math Processor optional
- 256K RAM on mother board expandable to 640K
- ROM 8K Bios
- 6 empty slots for expansion
- 2 serial ports one optional with expansion kit
- 1 parallel port
- 1 game port
- Clock calendar with software
- Hi-Res monographics video board
- Floppy controller
- Dual Floppy Drives 360K ea.
- 150-watt XT Power Supply
- AT style compatible keyboard
- Hi-Res TTL Green or Amber 12" monitor
- MS-DOS operating system and manual.
- Instructional Documentation and Utility Software
- Assembled and tested in America
- Optional internal 20 meg sub system for Super Turbo add $499

Special printer pricing with purchase of above computer.
MSO, the leader in cost effective computer systems, is offering the Sanyo MBC 1200 at a giveaway price!

The MBC 1200 Accounting Software -G/L, A/R, A/P, Inventory and Payroll modules retail for $398 per module. Now from MSO you pay only $99 per module or all five for $299! Second disk drive with fan-option available. COMPAT disk utility available.

SPECIFICATIONS:
• Two Z-80A (main and subsidiary) CPUs with no-wait mode for fast execution, substantial memory capacity (RAM 64KB, ROM 4KB).
• High-resolution full graphic function with 640 x 400 dot matrix display.
• Choice of 33 or 40 line text mode.
• CP/M operating system complete with interpreter, editor and all utilities.
• Easy-to-use Sanyo graphic BASIC.
• One (MBC 1200) or two (MBC 1250) internal double-sided, double-density, double-track, 5¼" slim-type 640KB formatted mini floppy disk drives.
• Special design featuring detachable ergonomic keyboard with coiled cable.
• Interfaces for Centronics printer and one RS-232C port provided.

FREE SOFTWARE
• Wordstar • Spellstar
• Calcstar • Mailmerge
• Infostar • Basic

CASH PRICE ONLY
Check in advance. Add 3% for VISA/MC. Shipping & handling charges will be added to each order.

For our catalog with complete details and prices, send $2 to:
Micro Supply Organization, Inc.
4909 Stockdale Hwy. #150
Bakersfield, CA 93309

15% Restocking on Returned Orders

MICRO CORNUCOPIA, #31, Aug-Sept 1986 21
I hate my blinking cursor. Sandy hates her blinking cursor. Larry and Gary hate their blinking cursors. With all that hatred around, it's easy to see why there's trouble in the world. You can do your part to make the world better by changing your cursor.

But, you can't do it in software. The little turkey is wired to blink, and blink it shall. Sigi, however, didn't realize that hardware is sacred (he has a lot to learn), so he stifled his cursor. Hopefully he'll stifle a few more things on the PC before they find out that he has been poking amongst the chips.

Hi! I just switched computer systems (for the first time ever), and it's great fun.

I've been using S-100 computers of different varieties and with different operating systems since 1977. Now that I work for a company that produces IBM PC coprocessor cards, I decided that now's the time to switch to something PC-compatible. Right now I'm using a PC Tech 80186 machine, which is really nice and super fast (almost as fast as an AT!!) but will get myself something more conservative to be able to plug in a V20 and run (nostalgia, yea!!!) some of my 8080 programs.

What I'm so long-windedly trying to get at is that I passionately hate blinking cursors — they drive me up the wall (cause headaches). Everyone I've talked to has told me, "That's the way it's designed, it's in hardware."

Surely it had to be done in hardware since the 6845 CRT controller can be programmed for two different blink rates — even non-blink! Unfortunately, these days it seems people are better with their fingers on the keyboard than on the chips.

I'm told even Peter Norton (who appears to be as much the IBM guru as Ward Christensen was the 8080/S-100/CP/M guru) said, "It couldn't be done." Now, are a few gates here and traces there that intimidating? Certainly not!

Unfamiliar Leads

Being a bit unfamiliar with TTL, having done no hardware work in a while (but catching up fast...), I was at first totally lost looking at the monochrome-graphics card I'm using (can't tell the brand — something Taiwanese I think). But after I got a look at a schematic (yes, some of the cheap Far East clones have tiny little schematics which are almost unreadable), I found the obvious solution to the blinking menace — a knife!

It appears as though most monochrome display cards (and graphics such as Hercules and "compatibles") use the same basic layout. This is roughly how it works, for those who have no schematics —

The CURSOR output (pin 19 of the 6845) goes into a 74LS174. There it's delayed (since the cursor location as output by the controller is normally at the location of the last character). The cursor is delayed by one character position, so it's displayed just beyond the last character. This is done in two flip-flops in the '174. The Q5 output is the delayed cursor signal. It's fed into a 7464, usually on pin 9.
The '64 consists of four AND gates feeding a NOR gate. Pins 9 and 10 are the inputs of a 2-input AND, and pin 10 is the blinkin' clock pulse.

To force this long story to an abrupt end, simply cut the trace leading to pin 10 of the 7464 and the cursor will quit blinking. You might want to do it right and pull pin 10 up to 5V via a 2.2K resistor after you've verified that it indeed works.

Don't Be Confused

There appears to be only one 7464 on the board, so you can't miss. Don't be confused by the board layout — the 64 may be clear on the other end of the board as seen from the 174! If your display card doesn't have a 7464, then you'll have to do some tracing. A schematic would be very helpful at this point.

In all the display cards I've seen so far, the cursor signal is delayed in a 74LS174 using two sections of the chip. The delayed cursor signal then enters an AND or NAND gate at one point.

If it's a 2-input gate, all you should have to do is CAREFULLY cut the trace leading to the other input. It is important to be able to reverse this procedure in case you cut the wrong trace. Beware, though, since traces may pass under chips and emerge at a different point on the other side, or even switch circuit board sides. The (N)AND gate may even be located at the other end of the board.

In tracing through a few more cards, I found an unnamed Taiwanese color graphics card where pin 19 of the 6845 goes to one section of a 74LS273 and from there to a 74LS00; cutting the other input to the '00 stopped the blinking.

More Info

Here is some more information, based on the IBM manuals:

1. Monochrome Display Adapter: cut trace going to pin 9 of U3 (LS08).

One problem with turning the cursor blink off this way is that if your cursor is something other than an underline (like a block), you can't see the character immediately below the cursor if you move it around the screen!

Controlling Your Cursor

You may use the following code fragment to control the cursor under DEBUG or include it in programs:

MOV AH, 1
MOV CX, cccc
INT 10H

Here are a few values for "cccc" for you to play with: (INT 10, subfunction 1 is explained in detail in various manuals and books, but no mention is made of bits 5 and 6 of register CH).

ccc = 600BH = slow blinking block
ccc = 400BH = fast blinking block
ccc = 000BH = steady block
ccc = 080BH = steady underline

As you can see, bits 6 and 5 of the CH register control blink:

<table>
<thead>
<tr>
<th>BIT 6</th>
<th>BIT 5</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>no blink</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>no cursor display</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>fast blink</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>slow blink</td>
</tr>
</tbody>
</table>

Poof!

I hope this has helped some people who, like myself, are suffering from blinking cursors. I might add that cutting traces on circuit boards makes your warranty vanish in a puff of greasy black smoke.

Fooling with electronic equipment not knowing what you're doing may make the equipment malfunction horribly, causing much grief and emptiness of the wallet, so you should attempt to unblink your cursor only if you are quite sure you know what you're doing. If all else fails ask a friend to do it for you. Just don't blame ME for breaking your computer, okay?

May this cure all your blinkin' cursors.

Editor's note: If the chips on your video card bear no relationship to anything living or dead, you might take a scope or logic probe and poke around the board. You'll be looking for a signal whose oscillation corresponds precisely with the blink of the cursor. Then pull out a pin or open a trace carrying that signal and see what happens. Again, as Sigi mentioned, you're on your own.
Game Theory Models In PROLOG And C:

Interfacing Between Languages

C and PROLOG are about as similar as IBM and Morrow, but that doesn’t mean they have to spend their lives playing oil and water. Here Gary takes a look at combining the two. (As far as I know, the first time in publishing history these two have been mated.)

The theory of games is a branch of mathematics first formulated by Von Neumann and Morgenstern in 1944. It’s been used in economics, evolutionary-, molecular-, and population-biology, linguistics, and physics to model interaction and conflict between variables (animals, genes, fluxes, particles).

Battles On Wildcat Mountain

For example, imagine a population of wildcats living on Wildcat Mountain in central Colorado. Let’s say there are distinct populations of three species of wildcats. Each cat within a population has its own den and territory, but will sometimes wander beyond its own lot to find food, water, etc.

If it encounters members of its own species while outside its own territory, no problem. The two cats exchange wags or whisker rubs (or whatever species while outside its own territory, fluxes, particles).

“Wildcat Battles” is obviously a very simple system, but it illustrates a method for programming more elaborate “game theories” in PROLOG. Let’s look first at a matrix of encounters below.

\[
\begin{array}{ccc}
X & case01 & case02 \\
Y & case03 & case04 \\
Z & case05 & case06 \\
\end{array}
\]

In cases _____, a wildcat has encountered a member of his own species, so there’s no ensuing population change. In each of the other cases, the populations of the two encountering species do change reflecting the outcome of a battle.

In case01, for example, a wildcat from population X has wandered out of his territory, encountered a Y, and lost the battle. X’s and Y’s populations will change, while Z’s will be unaffected.

Figure 1 shows the conditions for case(01) in Turbo PROLOG (Figure 2 shows the entire program with all cases). If you’re a C or Pascal programmer, think this way — case is a procedure or function called with the value parameter: 1.

It will execute, assigning values to variables X,Y,Z, and T, then pass those values to _tracker, another procedure (or predicate in PROLOG). The variables — A,B,C, and Tt — will contain values on return from _tracker.

Next, two predicates (times and equal0) are called to check the timer for the game. I’ve arbitrarily chosen to stop the Wildcat Battle after 50 encounters.

The remaining code in case(1) pertains to the user interface. I’ve chosen to monitor the battle with three windows representing population sizes and one window representing time.

In Turbo PROLOG you make and remove windows with the built-in predicates “makewindow” and “removewindow”. Change the window attribute with “window_attr”, and open a window for writing or drawing with “shiftwindow”. The predicate “readchar” halts execution until it reads a character from the keyboard. In case(1), any character (_) will be acceptable, and execution will resume. This allows me to step through the battle.

Two other predicates are important for the game (or model) — rand_1_9, which calls Turbo PROLOG’s built-in random generator (see The Last Page for a better random generator), and main, which controls the flow of the program.

Each encounter is preceded by a call to rand_1_9 which generates a random number between 1 and 9 corresponding to our nine possible cases. After each encounter, the timer and
arrays are updated and their values are displayed — the timer in a window at row = 16 and column = 60 and the values in the arrays as blocks (windows) in inverse video beginning at 2,2; 6,2; and 10,2. It's pretty flashy!

When we run the program, it begins executing at SETUP, the "GOAL" we specified at the end of the program. We could have specified any predicate as a goal (and consequently have started anywhere in the program). Sometimes, of course, it's better to begin in the beginning, but that's another story.

**Interfacing PROLOG To C**

In PROLOG, information is usually stored in a database. The database is built into standard PROLOG, but must be declared in Turbo PROLOG.

A database would be handy in large, mixed-data programs, but I wanted something simpler for my model — an array or structure. Unfortunately PROLOG has neither, nor an assignment statement equivalent to —

\[ X = X + 1 \]

or

\[ + + X \]

so I decided to let a C function handle the storage. See the figure below.

```c
int _actused;
int pop_a[2]={0,0};
int pop_b[2]={0,0};
int pop_c[2]={0,0};
int time[2]={0,0};

tracker_0(x,y,z,t,a,b,c,tt) int x;  
  int y;  int z;  int t;  int *a;  
  int *b;  int *c;  int *tt;
{}
  { pop_a[0]=pop_a[0] + x;
    *aspop_a[0];
    pop_b[0]=pop_b[0] + y;
    *bsspop_b[0];
    pop_c[0]=pop_c[0] + z;
    *ospop_c[0];
    time[0]=time[0] + t;
    *tt=time[0];
  }

The function is simple (with an exception or two) — Initialize four little arrays (one for each species of wildcats and one for time) to zero. When we call tracker_0, we pass it "x,y,z, and t'", and it returns new values for "a,b,c,tt" when it's finished executing. Each array (pop_a, pop_b, pop_c) keeps track of the population of one species.

You might have noticed a couple of oddities in the C function — the variable, _actused, and the function name itself, tracker_0. Both are required for a successful interface between Turbo PROLOG and Microsoft C (the C compiler I used).

The function, _actused, is generated by Microsoft C, which expects to have it resolved by one of its standard libraries at Linking. Unfortunately, Turbo PROLOG will currently allow linking only to its own library, PROLOG.LIB, so we have to handle any compiler-generated variables. In this case, there's only one, _actused, and we declare it to be an integer.

The underscore (_O) in tracker_0 is a cue used by Turbo PROLOG to recognize a function (or predicate) from another language. The "0" indicates that this "tracker" is the first function named "tracker" in the program. We could have multiple "trackers" named, "tracker_1", "tracker_2", etc.

**Programming in Turbo PROLOG**

Like Pascal, Turbo PROLOG is a strongly typed language. Databases, user-defined domains, and predicates are declared at the beginning of the program.

The C function (or any other function to be linked in from another language) must be declared as a global predicate, and its flow pattern must be specified —

```c
language c

  global predicates
  _tracker(integer,integer,integer,
          integer,integer,integer,
          integer) - (i,i,i,i,0,0,0)

Our flow pattern here states that four variables of type integer will be input to the C function (i,i,i,...), and four variables of type integer will be output from the C function (i,i,i,0,0,0).

**Working In Two Languages**

In order to interface the C function to our PROLOG program, we need to compile them separately (of course) to .OBJ files; then link them (along with several other files) to make an .EXE file. As you might expect, you need to follow a few rules to successfully create a standalone program.

First, compile the C function with the /AL (for large memory model) and /Gs (to remove the stack probe) options on the Microsoft C compiler. Case is important; "Gs" is not the same as "GS".

Then, choose the .OBJ compiler option in Turbo PROLOG when you compile WILDCAT.PRO. This will generate an .OBJ and a .SYM file.

Next, Link in all the files you need, using the /NOD option to eliminate stack checking —

```c
Link /NOD INIT + WILDCAT + TRACKER + WILDCAT.SYM,WILDC,PROLOG

WILDC will be the executable file; PROLOG is PROLOG.LIB; INIT, WILDCAT, and TRACKER are .OBJ files.

And that's it. We're done (maybe).

**Debugging**

Since the compile, link, execute sequence is so time consuming, I opted to utilize Turbo PROLOG's quickness when it compiles to memory by not making a .OBJ file until the program was reasonably clean.

I set up dummy predicates within the program to represent the interface to the C function. The program of course didn't do exactly what I wanted it to do, but it simulated it. Good enough for debugging. The compile, run, correct, and recompile sequence took a few seconds, instead of a lot of seconds, making debugging a pleasure.

In Figure 2 (the complete program) I've left in (between comment delimiters) the dummy function TRACKER and a case predicate, case(_), which satisfies any call to case in the program.

When I wanted to test the program quickly, I commented out the global predicate declaration for TRACKER and the complex case calls. When I wanted to test the interface, I commented out the dummy predicates (and their declarations).

In addition, Turbo PROLOG has a
GAME THEORY MODELS

(continued from page 25)

TRACE feature which allows you to step through the program while it's executing. Very nice!

Bye

My intention has been to give you an idea of the flexibility and power of PROLOG, in this case by using Borland's very user-friendly (and powerful) Turbo PROLOG and Microsoft's excellent C.

For more information about Turbo PROLOG contact —

Borland International
4585 Scotts Valley Dr
Scotts Valley CA 95066
(408) 438-8400

For more info about Microsoft C —

Microsoft Corp
Redmond WA
(800) 426-9400

Figure 2 - Beginning Of PROLOG Program

/* Program WILDCAT */

global predicates
_tracker(integer,integer,integer,
integer,integer,integer,integer,integer)
- (1,i,i,i,i,o,o,o) language c

predicates

/* _tracker(integer,integer,integer,
integer,integer,integer,integer,integer) */

rand_1_9(integer)
case(integer)
equalO(integer,integer,integer,
in integer, integer, integer)
times(integer)
setup
continue
main

clauses
setup:
- makewindow(4,7,1,"Time"),16,60,3),
main.

rand_1_9(X):=
random(Y), X = Y % 8 + 1.

main:-
rand_1_9(X), nl, case(X),
readchar(_,), continue.

continue:-
main.

/* case(_) :-
X = 1, Y = 2, Z = 3, T = 1,
_tracker(X,Y,Z,T,A,B,C,Tt), times(Tt),
equalO(A,B,C,A1,B1,C1),
shiftwindow(4), cursor(1,2), write(Tt),
readchar(_,), clearwindow,
makewindow(1,7,0,"",2,3,A1),
window_attr(112), removewindow,
makewindow(2,7,0,"",6,2,3,B1),
window_attr(112), removewindow,
makewindow(3,7,0,"",10,2,3,C1),
window_attr(112), removewindow.

_tracker(X,Y,Z,T,A,B,C,Tt):-
A=X, Y=B, Z=C, T=Tt.
*/

case(1) :-
X= -1, Y = 1, Z = 0, T =1,
_tracker(X,Y,Z,T,A,B,C,Tt), times(Tt),
equalO(A,B,C,A1,B1,C1),
shiftwindow(4), cursor(1,2), write(Tt),
makewindow(1,7,0,"",2,3,A1),
window_attr(112), readchar(_,),removewindow,
makewindow(2,7,0,"",6,2,3,B1),
window_attr(112), readchar(_,),removewindow,
makewindow(3,7,0,"",10,2,3,C1),
window_attr(112), readchar(_,),removewindow.

case(2) :-
X= -1, Y = 0, Z = 1, T =1,
_tracker(X,Y,Z,T,A,B,C,Tt), times(Tt),
equalO(A,B,C,A1,B1,C1),
shiftwindow(4), cursor(1,2), write(Tt),
makewindow(1,7,0,"",2,3,A1),
window_attr(112), readchar(_,),removewindow,
makewindow(2,7,0,"",6,2,3,B1),
window_attr(112), readchar(_,),removewindow,
makewindow(3,7,0,"",10,2,3,C1),
window_attr(112), readchar(_,),removewindow.

case(3) :-
X= 1, Y = -1, Z = 0, T =1,
_tracker(X,Y,Z,T,A,B,C,Tt), times(Tt),
equalO(A,B,C,A1,B1,C1),
shiftwindow(4), cursor(1,2), write(Tt),
makewindow(1,7,0,"",2,3,A1),
window_attr(112), readchar(_,),removewindow,
makewindow(2,7,0,"",6,2,3,B1),
window_attr(112), readchar(_,),removewindow,
makewindow(3,7,0,"",10,2,3,C1),
window_attr(112), readchar(_,),removewindow.
case(4) :-
X = 0, Y = -1, Z = 1, T = 1,
_tracker(X,Y,Z,T,A,B,C,Tt), times(Tt),
equal0(A,B,C,A1,B1,C1),
shiftwindow(4), cursor(1,2), write(Tt),
makewindow(1,7,0,"",2,2,3,A1),
window_attr(112), readchar(1),removewindow,
makewindow(2,7,0,"",6,2,3,B1),
window_attr(112), readchar(1),removewindow,
makewindow(3,7,0,"",10,2,3,C1),
window_attr(112), readchar(1),removewindow.
case(5) :-
X = 1, Y = 0, Z = -1, T = 1,
_tracker(X,Y,Z,T,A,B,C,Tt), times(Tt),
equal0(A,B,C,A1,B1,C1),
shiftwindow(4), cursor(1,2), write(Tt),
makewindow(1,7,0,"",2,2,3,A1),
window_attr(112), readchar(1),removewindow,
makewindow(2,7,0,"",6,2,3,B1),
window_attr(112), readchar(1),removewindow,
makewindow(3,7,0,"",10,2,3,C1),
window_attr(112), readchar(1),removewindow.
case(6) :-
X = 0, Y = 1, Z = -1, T = 1,
_tracker(X,Y,Z,T,A,B,C,Tt), times(Tt),
equal0(A,B,C,A1,B1,C1),
shiftwindow(4), cursor(1,2), write(Tt),
makewindow(1,7,0,"",2,2,3,A1),
window_attr(112), readchar(1),removewindow,
makewindow(2,7,0,"",6,2,3,B1),
window_attr(112), readchar(1),removewindow,
makewindow(3,7,0,"",10,2,3,C1),
window_attr(112), readchar(1),removewindow.
case(_):-
X = 0, Y = 0, Z = 0, T = 1,
_tracker(X,Y,Z,T,A,B,C,Tt), times(Tt),
equal0(A,B,C,A1,B1,C1),
shiftwindow(4), cursor(1,2), write(Tt),
readchar(1), clearwindow,
makewindow(1,7,0,"",2,2,3,A1),
window_attr(112), removewindow,
makewindow(2,7,0,"",6,2,3,B1),
window_attr(112), removewindow,
makewindow(3,7,0,"",10,2,3,C1),
window_attr(112), removewindow.
equal0(A,B,C,A1,B1,C1) :-
equal1(A,A1), equal1(B,B1),
equal1(C,C1).
equal1(A,A1) :-
A>X0, A1 = A.
equal1(A,A1) :-
A<X0, A1=A.
times(Tt):-
Tt<=50.
times(Tt):-
Tt>50, exit.

GOAL SETUP

End of WILDCAT
### EPIC SALES, INC.

#### FIXED DISK DRIVE SYSTEMS

<table>
<thead>
<tr>
<th>Model Type</th>
<th>10 MEG</th>
<th>20 MEG</th>
<th>30 MEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>TI PC (Zobex)</td>
<td>$699</td>
<td>$749</td>
<td>$899</td>
</tr>
<tr>
<td>TI PC (Western Auto.)</td>
<td>999</td>
<td>1049</td>
<td>1199</td>
</tr>
<tr>
<td>TI PC (SASI)</td>
<td>599</td>
<td>649</td>
<td>799</td>
</tr>
<tr>
<td>TI Bus. Pro. IBM PC and Compatibile*</td>
<td>749</td>
<td>799</td>
<td>949</td>
</tr>
<tr>
<td>Sanyo 550/555 Zenith Z-100</td>
<td>N/A</td>
<td>799</td>
<td>949</td>
</tr>
</tbody>
</table>

For External Units Add $200

Above include drive, controller, cables, one year warranty and 30 day return privilege. Call for pricing on larger size drives.

*Compatible: AT&T, Compaq, Tandy, Columbia, Corona, Zenith, Others.

#### TAPE BACKUP SYSTEMS

<table>
<thead>
<tr>
<th>Model Type</th>
<th>10 MEG</th>
<th>20 MEG</th>
<th>30 MEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Irwin Internal 10 Meg</td>
<td>$595</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Irwin Internal 20 Meg for AT</td>
<td>695</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wangtek Internal 60 Meg</td>
<td>1049</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For External Units Add $100

#### AT&T PERSONAL COMPUTER

<table>
<thead>
<tr>
<th>Model Type</th>
<th>10 MEG</th>
<th>20 MEG</th>
<th>30 MEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>6300, Mono, 640K, 2 Floppy, DOS, Basic</td>
<td>$1995</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6300, Mono, 640K, 1 Floppy, 20 Meg, DOS, Basic</td>
<td>2395</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6300 Plus, Mono, 640K, 2 Floppy, DOS, Basic</td>
<td>4295</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For High Resolution Color Add: 459

#### OLIVETTI PERSONAL COMPUTER

<table>
<thead>
<tr>
<th>Model Type</th>
<th>10 MEG</th>
<th>20 MEG</th>
<th>30 MEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mono, 640K, 2 Floppy, DOS, Basic</td>
<td>$1895</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mono, 640K, 1 Floppy, 20 Meg, DOS, Basic</td>
<td>2295</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For High Resolution Color Add: 459

#### IDS PERSONAL COMPUTER

<table>
<thead>
<tr>
<th>Model Type</th>
<th>10 MEG</th>
<th>20 MEG</th>
<th>30 MEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mono, 256K, 2 Floppy, P-Port, DOS</td>
<td>$1195</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mono, 640K, 2 Floppy, 20 Meg, P-Port, DOS</td>
<td>1795</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### FIXED DISK DRIVES FOR IBM AT AND TI BUSINESS PRO

<table>
<thead>
<tr>
<th>Model Type</th>
<th>10 MEG</th>
<th>20 MEG</th>
<th>30 MEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 Meg Seagate ST-225 (65 ms)</td>
<td>$439</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 Meg Seagate ST-4026 (40 ms)</td>
<td>739</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30 Meg Seagate ST-238 (65 ms)</td>
<td>589</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30 Meg Seagate ST-4038 (40 ms)</td>
<td>959</td>
<td></td>
<td></td>
</tr>
<tr>
<td>33 Meg Rodime RO-236E (55 ms)</td>
<td>$729</td>
<td></td>
<td></td>
</tr>
<tr>
<td>51 Meg Seagate ST-4051 (40 ms) (Bus. Pro. only)</td>
<td>1149</td>
<td></td>
<td></td>
</tr>
<tr>
<td>72 Meg CDC Wren (35 ms)</td>
<td>1949</td>
<td></td>
<td></td>
</tr>
<tr>
<td>119 Meg Maxtor 1140 (30 ms)</td>
<td>4595</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### OKIDATA PRINTERS

<table>
<thead>
<tr>
<th>Model Type</th>
<th>10 MEG</th>
<th>20 MEG</th>
<th>30 MEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 182-IBM, 120 CPS, 9 in. Carriage</td>
<td>$259</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model 192-IBM, 160 CPS, 6 in. Carriage</td>
<td>439</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model 193-IBM, 160 CPS, 15 in. Carriage</td>
<td>609</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### FLOPPY DISK DRIVES

<table>
<thead>
<tr>
<th>Model Type</th>
<th>10 MEG</th>
<th>20 MEG</th>
<th>30 MEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epson SD-521 DDS 5.25 in. Floppy Drive</td>
<td>119</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shugart SA-465 DDS 5.25 in. Floppy Drive</td>
<td>159</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Siemens FD 200 8 in. DDS 5.25 in. Floppy Drive</td>
<td>185</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### EXPANSION CARDS

<table>
<thead>
<tr>
<th>Model Type</th>
<th>10 MEG</th>
<th>20 MEG</th>
<th>30 MEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>STB Super RIO Plus II Multifunction Card (64K)</td>
<td>$259</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STB Graphics II Plus Card</td>
<td>305</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STB Chauffeur Mono Graphics Card</td>
<td>259</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STB Super I/O Plus Card</td>
<td>195</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STB Serial Card</td>
<td>95</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STB Parallel Card</td>
<td>79</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zobex ZX2T Controller Card for TI PC</td>
<td>335</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Western Digital WDXS Interface Card for IBM PC</td>
<td>169</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Taxan Model 555 Color Graphics Card</td>
<td>229</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Central Point Copy II PC Option Board</td>
<td>89</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microyse o'Clock Real Time Clock</td>
<td>59</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Western Automation S3 Multifunction Card (0K)</td>
<td>355</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Western Automation S3 Multifunction Card (OK)</td>
<td>545</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Western Automation Clock Option</td>
<td>125</td>
<td></td>
<td></td>
</tr>
<tr>
<td>512 K Ram Expansion Card for TI PC</td>
<td>259</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128K Ram Expansion Card for TI Bus. Pro.</td>
<td>169</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Epic 1 Meg Ram Expansion Card for TI Bus. Pro.</td>
<td>995</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Epic 2 Meg RamExpansion Card for TI Bus. Pro.</td>
<td>1595</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Epic 3 Meg Ram Expansion Card for TI Bus. Pro.</td>
<td>2195</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### MODEMS

- Hayes 1200B Internal (IBM or TI) ........................................ $399
- Prentice Com Port External (IBM or TI) ................................ $399

#### PC REPLACEMENT POWER SUPPLY

- 150 Watt Side Switch ....................................................... 109

#### INTEL MATH COPROCESSOR IC

- 8087-3 (5 MB) ............................................................... $150
- 8087-2 (8 MB) ............................................................ 175

#### RAM CHIPS

- 4154 (150ns) ..................................................................... 3.30
- 4125 (150ns) ..................................................................... 4.00

#### NEC REPLACEMENT PROCESSOR

- V-20 (5 MB 6088 REPLACEMENT) .......................................... $20
- V-30 (6 MB 6088 REPLACEMENT) ........................................ 35

#### TAXAN COLOR MONITORS

- Taxan Model 530 High Resolution Monitor ......................... 539
- Taxan Model 540 High Resolution Monitor ......................... 639

#### OTHER PRODUCTS

- World Media 5.25 in. DSDD Floppy Diskette ....................... $10
- Disk File 120, 5.25 in. Disk Storage ................................. 9
- A+B Switch Box .............................................................. 25
- Keytronic KB5151 Keyboard .............................................. 259
- Cabinet and Power Supply for (3) 5.25 in. Drives .......... 72

EPIC SALES INC.
132 WALNUT PLANO CENTER
GARLAND, TX 75042
TOLL FREE ORDER LINE: 1-800-223-EPIC
TECHNICAL INFO AND INQUIRIES: 1-214-272-5724

TERMS: CASH, CHECK, COD, MASTERCARD AND VISA ADD 3%
AMERICAN EXPRESS ADD 5%. TEXAS RESIDENTS ADD 8.25% SALES TAX.
Every day, VEDIT PLUS helps thousands of programmers, writers and engineers get down to business.

So why do people who could have ANY text editor prefer ours? For a lot of reasons, including:

- **CAPACITY**—With VEDIT PLUS, file size is never a problem. And virtual disk buffering simplifies editing of even the largest files.
- **FLEXIBILITY**—VEDIT PLUS lets you edit up to 37 files simultaneously. So you can cut and paste. Edit programs. Edit text. Even perform numerous search/replace functions in several files without user intervention.*
- **CUSTOMIZATION**—With VEDIT PLUS, you can create your own on-line editing functions with keystroke macros. Develop your own on-line help screens. Determine and revise your own keyboard layout easily.

- **SPEED**—VEDIT PLUS not only works hard, it works fast. Faster, in fact, than any other text editor on the market.
- **EXPERIENCE**—Six years ago, CompuView revolutionized the concept of microcomputer text editing. And we’ve been improving our products and services ever since.

Special Offer: Order a VEDIT PLUS text editor for $225 and we’ll include our V-PRINT™ document formatter—a $120 value—absolutely free.

Call CompuView today at 313/996-1299. You’ll be in good company.

VEDIT® Plus Text Editor

The Navy charts new concepts with it... GM engineers the future with it... National Can preserves facts with it... GE has bright ideas with it... Here's why you shouldn’t be without it.

CompuView®

CompuView® Products Inc., 1955 Pauline Boulevard—Suite 300, Ann Arbor, Michigan 48103, TELEX 701821

Available for PC DOS, MS-DOS, CP/M, CP/M-86.

*Free sort, compare, print and main menu macros included; optional 8080-8086 translator or mailmerge, $50 each.
Serial communications routines are pretty easy, once you get the hang of them. Unfortunately "getting the hang" isn't always easy. Greg wrote two serial communications programs, one for a CP/M Kaypro II, the other for a PC-compatible. This is a good example for those of you with data transfer problems.

In March of 1986, the company which publishes some of my software purchased a new Zenith Z-138 personal computer. It's a portable IBM PC-compatible with two 360K drives, 256K of RAM (which I helped them expand to the full 640K), and MS-DOS 3.0. I am very impressed with the Z-138's performance and price.

This publishing company already owned a Kaypro II with which they had generated megabytes of text and software. Unfortunately the company didn't have a disk-copying utility to convert from CP/M to MS-DOS or vice versa, so all the data was trapped in CP/M format.

So I wrote two programs (one for each computer) so I could transfer files back and forth via the serial ports. These two programs lend themselves to a wide variety of applications (networks, bulletin boards, modems, etc.) besides basic file transfer.

The programs are written for Turbo Pascal, version 2.0 (CP/M) and version 3.0 (MS-DOS). Except for minor changes, the CP/M 80 code compiles on the MS-DOS machine (which is, incidentally, why I needed this transfer program in the first place).

How the Operation Works
The basic operation is fairly simple. Any type of file can be transferred in either direction. You have your choice of four baud rates, and data is stored on disk.
PROCEDURE LogOn;
BEGIN
  ClrScr;
  writeln('File Transfer Utility Program -- Version 1.0');
  writeln('for Zenith Z-130 and IBM PC-Compatibles');
  writeln('running under MS-DOS 3.1');
  writeln;
  writeln('Copyright (c) 1986 by Greg C. Flothe');
  writeln('All Rights Reserved');
  Delay(3000);
END; {LogOn}

PROCEDURE InitPort;
BEGIN
  ah:= 0;
  al:= BaudByte shr 5 + $03;
  WITH recpack DO
    ax:: ah shl 8 + al;
    dx:= 0;
    intr($14, recpack); {interrupt & change serial port parameters}
  writeln('Serial Port Ready');
END;

PROCEDURE BaudRate; {establish serial port speed with code}
VAR
  Baudtype: integer;
BEGIN
  writeln('Baud Rate currently at ', Baud);
  write('Change rate? '); readln(Response);
  IF UpCase(Response) = 'Y' THEN
    BEGIN
      write('Enter 1>300 2>1200 3>4800 4>9600: '); readln(BaudType);
      CASE BaudType OF
        1: BEGIN
            Baud:= 300;
            BaudByte:= BaudCode300;
          END;
        2: BEGIN
            Baud:= 1200;
            BaudByte:= BaudCode1200;
          END;
        3: BEGIN
            Baud:= 4800;
            BaudByte:= BaudCode4800;
          END;
        4: BEGIN
            Baud:= 9600;
            BaudByte:= BaudCode9600;
          END;
      END;
    END;
  END; {if}
  initport; {send Baud code to serial port}
  writeln('Baud Rate set to ', Baud, ' BPS.');
END; {BaudRate}

PROCEDURE SetUpIO; {Set Input/Output speed, flow}
BEGIN
  ClrScr;
  BaudRate;
  writeln; write('I/O MODE - '); CASE Mode OF
    send: writeln('TRANSMIT');
    receive: writeln('RECEIVE');
  END;
END; {SetUpIO}

Sending

The procedure SendHeader initiates the transfer process by sending an SOH (start of header) character (01). The transmitting machine then waits for a verifying echo.

When it receives the verification, the transmitter sends the low-order byte of the number of records to be transferred. If that byte is echoed correctly then it sends the high-order byte. When this byte is echoed properly, the transfer begins.

(Note: I used a two-byte block count so I could transmit files of up to 64K bytes. A single byte would limit me to 255 blocks (X 128 = 32K bytes).)

After the block count has been verified, I don't do any further error checking, although each block could be verified by a checksum. There are numerous ways to handle error checking, but in a direct connect situation, errors are seldom a problem.

Receiving

The receiving system asks for a file name and then it opens the file. Procedure GetHeader then echoes the SOH, and the low and high bytes of the block count (file size).

Then it simply receives data until the number of blocks received equals the block count.

Hard Wire

These programs can easily be adapted for use with modems, although our application was hard-wired, using 25-pin 'D' connectors, one male (Kaypro) and one female (IBM, Zenith). Figure 1 shows how the two computers were connected.

Only four wires are required, although a fifth (shield wire) was also used in our case. The cable should be two twisted pairs (shielded, optionally) of solid #22 or #24 (telephone) wire. Note that one conductor from each pair ties to pin 7 (common); the other wire carries the signal.

(Editor's note: For cables under 50 feet, 3 wires should be sufficient. Just connect pin 7 to pin 7, pin 2 to pin 3, and pin 3 to pin 2.)

Be especially careful when wiring these connectors; it's easy to count the pins backwards (think of mirror im-
ages and you’ll see the problem).

From the back, on the male (Kaypro) end, pin 1 is on the upper right; on the IBM/Zenith end, pin 1 is on the upper left. Plugging the connectors together may help to visualize their opposite relationship.

**CPMTRANS.PAS**

I started with the CP/M version because it was easier. The Kaypro II uses three ports to control serial communications: Port[0] for baud rate control, Port[4] for data input and output, and Port[5] for reading port status. Since Turbo Pascal can read and write to ports in the Port array, I declared these port addresses as constants.

Similarly, the baud rate codes are also declared constants. Any of these constants may be changed to fit the serial port address or baud rate codes of another machine, of course.

To detect a received character I test bit 0 of StatPort (Port[5]) by ANDing the byte from that port with hex 01. WaitForChar handles this task. To see if a character has been sent, we test bit 2 of StatPort by ANDing its byte with hex 04. WaitForSend does this.

If the result of the ANDing is greater than 0, then the condition is TRUE. Otherwise I stay in the loop. You can interrupt the loop (in case of an error, etc.) by pressing a key.

The byte at DataPort (Port[4]) is either our received character (IN or read operation), or transmitted character (OUT or write).

**File Handling**

To streamline things in file handling, I elected to use the BlockRead/BlockWrite commands, and to read each block into a buffer. The buffer is an array with a record size of 128, which is the value of constant RecSize. Procedures InBlock and OutBlock receive or send a block of buffer data from/to the serial port.

If the file is a text file, printout is suppressed from the first Ctrl-Z (end-of-file marker) onward if the last block is being transferred. That way you won’t see garbage characters on the screen.

No check is made to ensure that the file to be printed is indeed printable,
PROCEDURE InBlock;
BEGIN
Bytecount := 1;
WHILE Bytecount <= RecSize DO {read a block from port}
BEGIN
InChar(NewChar); {get char}
Buffer[Bytecount] := NewChar; {store it}
OutChar(NewChar); {echo char}
IF PrintOn THEN
BEGIN
IF ((Remaining = 1) AND (NewChar = 26)) THEN
PrintOn := false {search for AZ (EOF) to halt output}
ELSE
write(Char(NewChar));
END;
Bytecount := succ(Bytecount);
END; {while Bytecount}
END; {InBlock}

PROCEDURE ReceiveFile; {get a file from ser. port & store it}
BEGIN
writeln; writeln(‘Name of file to be received? ’);
readln(FileName);
writeln;
IF FileName <> ‘’ THEN
BEGIN
Assign(Dest, FileName); {open file for write}
Rewrite(Dest);
writeln;
write(‘Incoming File Ready (Y/N)? ’); {wait for cue}
readln(Response);
IF UpCase(Response) = ‘Y’ THEN
BEGIN
GetHeader;
writeln;
Str(Remaining:5, RemBlks); {turn Remaining into a string}
writeln(‘Blocks to be transferred: ’, RemBlks); {print it}
writeln;
PrintOn := PrintEnable; {send copy to screen if desired}
WHILE Remaining > 0 DO
BEGIN {Remaining is # of blocks to be read}
InBlock;
BlockWrite(Dest, Buffer, 1); {save complete record to disk}
Remaining := pred(Remaining);
END; {while Remaining}
close(Dest);
writeln;
writeln(‘File ’, FileName, ‘ written to disk.’);
END; {if Response}
END; {if FileName <> ‘’}
ELSE writeln(‘Aborting RECEIVE procedure.’);
END; {ReceiveFile}

PROCEDURE SendHeader;
BEGIN
NewChar := SOH; {Send Start-Of-Header char.}
OutChar(NewChar);
REPEAT
InChar(NewChar);
UNTIL KeyPressed OR (NewChar = SOH); {wait for echo}
NewChar := Lo(Remaining);
OutChar(NewChar); {Send low-order byte of Remaining}
REPEAT
InChar(NewChar);
UNTIL KeyPressed OR (NewChar = Lo(Remaining)); {wait for confirm.}
NewChar := Hi(Remaining);
OutChar(NewChar); {High-order byte to serial port}
REPEAT
InChar(newChar);
UNTIL KeyPressed OR (NewChar = Hi(Remaining)); {wait for confirm.}
END; {SendHeader}

PROCEDURE OutBlock;
BEGIN
Bytecount := 1;
WHILE Bytecount <= RecSize DO {Send a block to serial port}
{code continued on next page}
(continued next page)
for a service routine code. If a "0" is found there, a port parameter change is requested, and the "coding byte" in low AX is written to the serial port control byte.

Service routine "0" sets up the port parameters (speed, parity, stop bit(s), and character size).

We take our baud codes, shift them left 5 bits to positions 7, 6 and 5, and then add hex 03. Hex 03 (bits 0 - 4) tells the system that we want no parity, one stop bit, and an 8-bit character.

This baud rate (plus everything else) byte, gets stuck in the lower half of the AX register and gets sent out when the high byte of AX is 0 (that's service routine "0").

More Interruptions
Input and output with interrupts is almost the same, since all port operations are handled by interrupt 14. To test the port status (Procedure TestPort), a code of "03" is put in high AX, and the interrupt call returns with a 16-bit status word in AX which, among other things, tells if a character has been received (bit 8 set) and if a transmitted character has cleared. (bit 13 set).

The OutChar procedure of MSTRANS, after testing for the (TBE) transmit buffer empty signal (ANDed with StatWord = $2000) loads the high AX byte with the "output character" code "01", puts the character in low AX, and sends it via interrupt 14.

The InChar procedure tests for a received character available signal (ANDed with StatWord = $100), then loads the high byte of AX with the "input character" code "02", and calls interrupt 14, which returns the character in the low half of AX.

The rest of the procedures are very similar to the CP/M version. Each file transfer begins with SendHeader, which sends the SOH start-of-header character, its verifying echo, and the number of blocks in two bytes. Get-Header reads those three bytes, and echoes them for verification. Once the transfer process has begun, it will continue automatically until all the blocks have been transferred.

MSTRANS.PAS (continued from page 33)
BEGIN
NewChar:= Buffer[Bytecount];
OutChar(NewChar);
IF PrintOn THEN
BEGIN
IF (Remaining = 1) AND (NewChar = 26) THEN
PrintOn := false
ELSE
write(Char(NewChar));
END;
InChar(NewChar);
Bytecount:= succ(Bytecount);
END;
END; {OutBlock}

PROCEDURE SendFile; {get an MS-DOS file and transfer it}
BEGIN
writeln;
REPEAT
writeln;
write('Transfer from file name: ');
readln(FileName);
assign(Source, FileName);
reset(Source) {$1+};
OK:= (IOresult = 0);
IF NOT OK THEN
writeln('Cannot find file ',FileName);
UNTIL (OK = true) OR (FileName = '');
IF OK THEN
BEGIN
Remaining:= FileSize(Source);
writeln('File ',FileName,' contains ',Remaining,' records.');
writeln;
SendHeader;
PrintOn := PrintEnable;
WHILE Remaining > 0 DO
{send 1 block at a time until done}
BEGIN
BlockRead(Source, Buffer, 1);
OutBlock;
Remaining:=pred(Remaining);
END;
writeln;
writeln('File ',FileName,' transferred.');
close(Source);
END {if}
ELSE
writeln('Aborting SEND procedure.');
END; {SendFile}
BEGIN {main program begins here}
LogOn;
Baud:=1200; {set up default parameters -- 1200 Baud, Receive Mode}
BaudBytes:=BaudCode1200;
Mode:= receive;
REPEAT
Setupio;
REPEAT
writeln('If this is a TEXT file, would you like the file?');
readln(Response);
IF UpCase(Response) = 'Y' THEN
PrintEnable:= false {disableable screen output}
ELSE
PrintEnable:= true;
END {if}
ELSE
writeln('Change Parameters, (<N> to exit)? ');
readln(Response);
UNTIL UpCase(Response) = 'N';
writeln('TRANSFER program done.');
END. {Transfer}

End of MSTRANS.PAS
I hope this information has introduced the basics of serial communications to those of you who are curious or might like to try it yourself.

References


XT COMPATIBLE HARD DISK SYSTEM SPECIAL  
$1259.00

XT-type motherboard (4.77 mhz) with 256K ram expandable to 640K on board. 10MB hard disk with controller, single Teac floppy with controller, high resolution amber monitor with Hercules compatible monographs card (720 x 348) and parallel printer port, AT-style keyboard (large return, shift, ctrl and tab keys; lighted caps-lock and num-lock indicators), flip-top case, and 135 watt power supply. Aw, what the heck, we’ll also throw in a parallel printer cable, assembly and burn-in.

Upgrades:  
Add-on  
4 layer 8mh Turbo  
(fast, reliable)  
$129  
1200 baud modem  
(Hayes compatible internal, with software).  
$179  
640K memory  
$69  
20 MB hard disk  
$189  
Extra Teac floppy  
$109  
Battery clock, serial and game ports  
$50  

XT Compatible Floppy System Special  
$829.00

Same as above but with two Teac floppies and no hard disk.

CARDS & COMPONENTS

<table>
<thead>
<tr>
<th>Component</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>XT-type motherboard (2 layers)</td>
<td>$129</td>
</tr>
<tr>
<td>8mh Turbo board (4 layers)</td>
<td>$229</td>
</tr>
<tr>
<td>Multi I/D card</td>
<td>$95</td>
</tr>
<tr>
<td>Battery clock, serial, parallel, game ports.</td>
<td>$95</td>
</tr>
<tr>
<td>Monographs card (Hercules compatible, includes printer port)</td>
<td>$95</td>
</tr>
<tr>
<td>Color/graphics card</td>
<td>$69</td>
</tr>
<tr>
<td>Case - flip top</td>
<td>$49</td>
</tr>
<tr>
<td>Power Supply (135W)</td>
<td>$89</td>
</tr>
<tr>
<td>20 Meg Hard Disk (XT)</td>
<td>$489</td>
</tr>
<tr>
<td>20 Meg Hard Disk (AT)</td>
<td>$650</td>
</tr>
<tr>
<td>Amber TTL Monitor (hi-res monochrome)</td>
<td>$110</td>
</tr>
<tr>
<td>Amber Monitor (composite with tilt and swivel)</td>
<td>$95</td>
</tr>
<tr>
<td>TVM Color monitor - RGB</td>
<td>$395</td>
</tr>
<tr>
<td>XT Assembly Manual</td>
<td>$49</td>
</tr>
</tbody>
</table>

WHY 4 LAYERS?

2 layer boards have traces on the top and bottom of the circuit board. 4 layer boards have two extra layers of traces sandwiched between the top and bottom of the board. When comparing a 2 layer board against a 4 layer board of the same size, the 2 layer board will have longer trace runs (which increases the resistance to the signals traveling through the traces) and the traces will be closer together (which increases the possibilities of interference from a neighboring trace). The faster the signals travel through the traces, the more exaggerated these problems become. For those who want to run an XT-type motherboard at 8mh, we recommend a 4 layer board.

WARRANTY

All products are warranted for 90 days.

AT BASIC

- 6/8 MHz Motherboard
- 512K RAM
- 1.2 Floppy Drive
- Western Digital H/F Controller
- AT Keyboard
- 200W Power Supply

$1889.00

AT SYSTEM

- 6/8 MHz compatible
- 512K RAM
- TTL Monochrome Monitor
- TTL Monochrome Board
- 20MB Seagate • 1.2 Floppy
- Western Digital H/F Controller
- Keyboard • 200W • Case

$2499.00

TURBO PC/XT

The Turbo PC/XT features an enhanced version of th industry standard Intel 8088 microprocessor, the 8088-2. In the normal mode, the 8088-2 runs at 4.77MHz. When more power is needed, a simple software command can put the Turbo PC/XT in a high speed mode of 8MHz. Performance is 68% greater than that of the IBM PC and other PC compatibles. For added user convenience, the high speed mode is clearly indicated by an easily visible green LED on the box.

For added effectiveness, the Turbo PC/XT offers powerful hardware and advanced software features to provide a multiple “virtual console” environment. This environment allows multitasking and windowing. Now, users have the ability to accomplish several tasks more quickly than ever before.

The Turbo PC/XT can run on both the Concurrent DOS and MSDOS operating systems. Thus, there is a high degree of compatibility with the extensive library of software designed for the IBM PC. Utilizing the latest in microprocessor technology, the Turbo PC/XT can deliver stronger performance at higher speed without sacrificing software compatibility. The Turbo PC/XT is the ideal computer for OEM’s, VAR’s and end users who demand personal computer performance together with PC reliability at an economical price.

McTEK SYSTEMS, INC.  
2316 FOURTH STREET  
BERKELEY, CA 94710  
(415) 549-3472

DEALERS
Identify yourself as a dealer and we’ll send you dealer information. We’re reliable, efficient and price competitive.
PROCEDURE SetUpIO;  {change input/output parameters}
BEGIN
   ClrScr;
   BaudRate;
   writeln('I/O MODE - ');
   CASE Mode OF
   send: writeln('TRANSMIT');
   receive: writeln('RECEIVE');
   END;
   writeln('Change Mode (Y/N)? ');
   readln(Response);
   IF UpCase(Response) = 'Y' THEN
   BEGIN
      writeln('THIS terminal in SEND or RECEIVE mode? ');
   END;
   writeln;
END; {SetUpIO}

PROCEDURE WaitForChar;
BEGIN
   REPEAT
      OK:= (Port[StatPort] AND $01) = 1; {wait for char.}
   UNTIL KeyPressed OR OK;
END; {WaitForChar}

PROCEDURE WaitForSend;
BEGIN
   REPEAT
      OK:= (Port[StatPort] AND $04) = 0; {ok to transmit?}
   UNTIL KeyPressed OR OK;
END; {WaitForSend}

PROCEDURE InBlock;  {read a block from serial port}
BEGIN
   Bytecount:= 1;
   WHILE Bytecount <= RecSize DO
   BEGIN
      WaitForChar;
      Buffer[Bytecount]:= Port[DataPort]; {read char. from port}
      WaitForSend;
      Port[DataPort]:= Buffer[Bytecount]; {echo character to port}
      IF PrintOn THEN
      BEGIN
         IF ((Remaining = 1) AND (Buffer[Bytecount] = 26)) THEN
         BEGIN
            PrintOn:= false {search for ~z (EOF) to halt output}
         ELSE
            write(Char(Buffer[Bytecount]));
         END;
      END;
      Bytecount:= succ(Bytecount); {increment byte pointer}
   END; {while bytecount}
END; {InBlock}

PROCEDURE GetHeader;  {Set up incoming file for transfer}
BEGIN
   REPEAT
      UNTIL KeyPressed OR (Port[DataPort] = SOH); {test for SOH character}
      Port[DataPort]:= SOH;
      WaitForChar;
      Remaining:= Port[DataPort]; {read low remaining record count}
      Port[DataPort]:= Remaining; {echo it}
      WaitForChar;
      Remaining:= Port[DataPort]; {read high remaining rec. count}
   END; {GetHeader}

PROCEDURE ReceiveFile;  {read a file from serial port and write to disk}
BEGIN
   writeln;
   writeln('Name of file to be received? ');
   (code continued on next page)
C-BUNDLE  $99

VIEW: CRT Based Disk Diagnostic
EZZAP: ROM Burning Utility
includes schematic
C-GAMES: User Modifiable Maze Game
All are written in C, include Source Code, and available separately.

NEW INTEGRATED BIOS FOR BIGBOARD II
This BIOS adds special features to your Bigboard II floppydisk and/or Winchester system.
Read, write and format any 5½ and/or 8½ disk formats. 40 formats included. Use any type of 5½ and/or 8½ drive. New SYSGEN works between 5½ and 8½.

INCLUDES:
• CONFIGuration program that lets you install any new floppy disk format INTERACTIVELY.
• FORMATTER allows you to format almost any diskformat.
• PC-COPY reads and writes PC diskettes on your Bigboard II.
• MONITOR EPROM with serial keyboard and translate table.
• 300 page ZCPR2 manual.
• 60 page BIGBOARD II tech manual.
• BOOTABLE DISK contains free ZCPR2 and P2DOS system. With TIME and DATE stamping, BIOS also has provisions for 256K RAM disk, Centronics, system in ROM.
• WINCHESTER FORMATTER and SYSGEN supports XEBEC, W-D and ADAPTEC type controllers. Subdivides into any specified number of drives.

Price: $99.95 (specify disk)

CPMTRANS.PAS (continued from page 37)

readln(fileName);
writeln;
IF FileName <> ' ' THEN BEGIN
assign(Dest,FileName);
Rewrite(Dest);
write('Incoming file ready? ');
readln(Response);
IF UpCase(Response) = 'Y' THEN BEGIN
GetHeader; {Wait for SOH char., read # of blocks remaining}
writeln;
Str(Remaining:5,RemBlks); {convert Remaining to 5-digit string}
writeln('Blocks to be transferred: ',RemBlks);
writeln;
PrintOn:= PrintEnable; {turn on display if enabled}
WHILE Remaining > 0 DO END; {while remaining}
InBlock;
BlockWrite(Dest,Buffer,1); {write to new file on disk}
Remaining:= pred(Remaining);
END; {while remaining}
close(Dest);
writeln('
File ',FileName,' written to disk.');
END; {if}
END; {ReceiveFile}

PROCEDURE OutBlock; {send a block of data to serial port}
BEGIN Bytecount:= 1;
WHILE Bytecount <= RecSize DO BEGIN
WaitToSend;
Port[DataPort]:= Buffer[Bytecount]; {send byte}
WaitForChar;
BufByte:= Port[DataPort]; {read echoed character}
IF PrintOn THEN BEGIN
IF ((Remaining = 1) AND (BufByte = 26)) THEN
  writeln(False {test for 'Z' (EOF character)})
ELSE
  write(Char(BufByte));
END;
Bytecount:= succ(Bytecount);
END;
END; {OutBlock}

PROCEDURE SendHeader;
BEGIN Remaining:= FileSize(Source); {get # of records to transmit}
writeln; writeln('File ',FileName,' contains ',Remaining,' records.');
Port[DataPort]:= SOH; {send start-of-header} REPEAT UNTIL KeyPressed OR (Port[DataPort] = SOH); {wait for echo}
Port[DataPort]:= Lo(Remaining); {send low block count}
REPEAT UNTIL KeyPressed OR (Port[DataPort] = Lo(Remaining));
{wait for verify}
Port[DataPort]:= Hi(Remaining); {send high block count}
REPEAT UNTIL KeyPressed OR (Port[DataPort] = Hi(Remaining));
{wait for verify}
END; {SendHeader}

PROCEDURE SendFile; {send file to serial port}
BEGIN writeln;
REPEAT writeln;
write('Transfer from file name: ');
readln(fileName);
assign(Source,FileName);
{$!-} reset(source) {$!+};
OK:= (IOreault=0);
IF NOT OK THEN writeln('Cannot find file ',FileName);
UNTIL (OK = true) OR (FileName = '');
$\text{IF OR THEN}\n\text{BEGIN}\n\text{SendHeader};
\text{PrintOn:= PrintEnable; \{turn on screen display\}}
\text{WHILE Remaining > 0 DO} \n\text{ BEGIN}\n\text{BlockRead(Source, Buffer, 1); \{get a block from disk\}}
\text{OutBlock; \{send it to serial port\}}
\text{Remaining:=pred(Remaining); \{until Remaining = 0\}}
\text{END;}
\text{writeln; writeln('File ',FileName, ', transferred.');}
\text{close(Source);} \n\text{END;} \{if\}
\text{ELSE}
\text{writeln('Aborting SEND procedure.');}
\text{END;} \{SendFile\}
\text{BEGIN \{Transfer\} \{main program begins here\} BEGIN \{Transfer\}
\text{port:= BaudCode1200; \{set up 1200 baud rate, receive mode\}}
\text{Mode:= receive; \{Default Mode = receive\}}
\text{LogOn;}
\text{REPEAT}
\text{SetUpIo;}
\text{REPEAT}
\text{writeln('If this is a TEXT file, would you like the file?');}
\text{readln(Response);}
\text{IF UpCase(Response) = 'N' THEN}
\text{PrintEnable:= false \{disable/enable screen output\}}
\text{ELSE}
\text{PrintEnable:= true;}
\text{IF Mode = send THEN}
\text{SendFile}
\text{ELSE ReceiveFile;}
\text{writeln;}
\text{write('Transfer another file (Y/N)? ',1);}
\text{readln(Response);}
\text{UNTIL UpCase(Response) = 'N';}
\text{write('Change Parameters, (Y to exit)? ',1);}
\text{readln(Response);}
\text{UNTIL UpCase(Response) = 'N';}
\text{writeln; writeln('TRANSFER program done.');}
\text{END. \{Transfer\} \{End of CPMTRANS.PAS\} END. \{Transfer\} \{End of CPMTRANS.PAS\}}

---

**5¼'' HARD DISK CONTROLLERS**

These new factory sealed Shugart controllers have manuals and schematics available. They will control up to two 5¼'' hard disk drives with up to 16 heads each. These unique controllers will mount directly on the drive. By the change of an EPROM they change their instruction set to emulate other popular controllers.

* Shugart 1610-3 emulates Xebec S1410
  * Apple II, II+, IIIE
  * MAC (see Sept. Dr. Dobbs)
  * *Wavemat* Bullet
  * All AMPRO boards (little board, etc.)
  * IS1 5160 & ACS 1000 (PC clones)

1 for $99. ea. 2 for $90. ea. 100 for $75. ea. Manual and schematic available.

* Computer Surplus Store (408) 280-1746
  * Drives 8'' SMD NEC O2246, FUJITSU 2312, 2322, MITSUBISHI 2360
  * 8'' SA1000 - QUANTUM Q2020, Q2030, Q2040, OMTI 20C, 20L

---

* Byte Magazine called it.*

"CIARCIA'S SUPER SYSTEM"

---

**The SB180 Computer/Controller**

Featured on the cover of Byte, Sept. 1985, the SB180 lets CP/M users upgrade to a fast, 4'' x 7 ¼'' single board system.

- 6MHz 84180 CPU
- 256K RAM
- 8K Monitor ROM with device test, disk format, read/write.
- Mini/Micro Floppy Controller
- (1-4 drives, Single/Double Density, 1-2 sided, 40/72/100 track 3½'' 5¼'' and 8'' drives).
- Measures 4'' x 7 ¼'', with mounting holes
- One Centronics Printer Port
- Two RS232C Serial Ports
- Auto-baud rate select.
- 

---

**SB180-1**

SB180 computer board w/256K bytes RAM and ROM monitor

---

**SB180-1-20**

same as above w/ZCPR3, ZRDOSS and BIOS source

---

NEW

**COMM180-M-S**

optional peripheral board adds 1200 bps modem and SCSI hard disk interface.

---

TO ORDER

TELEX

1-800-635-3355

643331

For technical assistance or to request a data sheet, call:

1-203-871-6170

---

**Micro Cornucopia, #31, Aug-Sept 1986**
Running Programs From Other Programs

Laine's back to his balcony and Turkish cherry juice. Laid back, feet propped up, he relaxes from the tribulations of his far eastern trip with a detailed description of how to call a program from within another program. Light and easy reading, this isn't — but interesting reading, this is.

I find it convenient, if not necessary, to execute one program while within another. I have two uses for external execution. The first is to call a program from a "mother" program to help mother do its job. The second is to temporarily escape what I'm doing to run another (possibly unrelated) program.

An example of each:

1. I have a backup program (I call FORMAT) that has to erase all the files on a backup disk before backing up files.
2. From the command line of the EXPRESS editor I let the user execute any DOS command. 

Turbo Pascal

I've written this subroutine for three languages on two operating systems (see Micro C #28 for the Pascal CCP/M-86 version), but the first language I wrote it in was Pascal. After looking in the Turbo Pascal manual you'll probably say, "But why did you do that? Turbo already has an 'Execute' procedure, you dipstick." Sure it does, but read the fine print. It only executes other TURBO programs, you can't pass command line parameters, and it doesn't CALL the new program, it JUMPS to it. Yuch.

Calling Exec

Now that we have justification, what do we have to do? First, how do we want to call it? The simplest way would be like this:

ErrNum := Exec(CommandLine);

where CommandLine is any string and Errnum is an integer that returns a success/failure indication. So, given a command string, how do we execute it? We need to send the command string to DOS function 4Bh, "Execute Program". The way to do this in Turbo Pascal is with the built-in "MsDos()" procedure. We'll just put pointers to the command line (and a few other things explained in a minute) in software registers (actually a record of type REGISTER) and execute "MsDos(reg)". Simple enough.

What MS-DOS Needs

If you look up function 4Bh in your DOS Programmer's Manual (you have been looking at your manual, haven't you?) you'll see that DOS wants two pointers (in registers DS:DX and ES:BX). The first they call the "path" of the command, and the second is a parameter block. The PATH is actually the command that you want to execute. The parameter block contains a pointer to the "command line trailer," default FCBs for the trailer, and an

```pascal
{----------------- test program for Exec -------------------}
PROGRAM TestExec;
{$PASEXEC.INC} CONST Command : Array [1..5] of STRING128 = ('dir/w','chkdsk','e','set','debug'); VAR Choice : CHAR; ct, ErrNum : INTEGER;
begin REPEAT FOR ct := 1 to 5 DO writeln(ct,'',Command[ct]); writeln; write('Enter your choice or <ESC> to quit: '); REPEAT read(kbd,Choice); UNTIL (Choice in ['1'..'5','A']); IF (Choice <> '') THEN begin writeln(Command[ord(Choice)-ord('0')]); ErrNum := Exec(Command[ord(Choice)-ord('0')]); IF (ErrNum <> 0) THEN writeln ('Error #1,ErrNum); end; { choice <> escape } UNTIL (Choice = 'A'); writeln('Goodbye'); end. { TestExec }
```
"environment pointer." A diagram would ease the confusion a bit: ('->' means 'is a pointer to')

<table>
<thead>
<tr>
<th>AH</th>
<th>4Bh</th>
<th>Exec function call</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>0</td>
<td>'load and execute'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DS:DX</th>
<th>--&gt;</th>
<th>name of command to execute</th>
</tr>
</thead>
<tbody>
<tr>
<td>ES:BX</td>
<td>--&gt;</td>
<td>aa (word) --&gt; segment of environment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bb:bb (dword) --&gt; command trailer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cc:cc (dword) --&gt; first default FCB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dd:dd (dword) --&gt; second default FCB</td>
</tr>
</tbody>
</table>

**Break Time**

I was just thinking about how wonderful life can be sometimes. I mean, here I am typing away on this column. One year ago I would have been in some hot sweaty room somewhere; 10 years ago I would have been using a typewriter; instead, here I am lying on a balcony on a couple of cushions listening to Rick James and drinking cherry juice with my little Toshiba T1100 portable PC on my lap and my toes wiggling in the Anatolian sunshine. My oh my....

Just figured out why the writing still isn't going so great. It's all that damn cherry juice. Just switched to Efes Pilsen and Little Stevie Wonder.

A note added later: know what wiggling my toes in the sunshine got me? One sunburned foot, that's what. The other one was in the shade.

**Back At The Ranch**

The way Microsoft has set up function 4Bh we could actually execute the program by grabbing the first name out of the command line and putting it in the PATH string, then parsing out the two FCBs and the command trailer. However, there are a few complications with doing it this way.

1. You have to parse the filenames in the command trailer and put them into FCB format yourself.
2. Since the PATH must have filename AND extension, you have to retry the Exec with EXE if COM doesn't work.
3. You must do BAT file processing yourself if it's to be done at all.
4. You don't have access to commands that are internal to COMMAND.COM.
5. You have to handle IO redirection yourself.

To make life easier, I've adopted the method of always using COMMAND.COM as the PATH. This way my Exec procedure is much less cluttered, I can execute BAT files, I don't have to parse the FCBs (COMMAND doesn't use the default FCBs), IO redirection is automatic, and I can use all the commands internal to COMMAND (like TYPE, DIR, etc.).

Using COMMAND.COM has its drawbacks, too. The most serious is that calling up COMMAND.COM an extra time uses about 15K more memory and extra disk access time. Also, you must always have a disk with COMMAND.COM in the drive. Since most of us have 640K anyway, I don't think the 15K extra is much problem, and the extra loading time isn't noticed at all if you have a winchester or a small RAM disk. Always having COMMAND.COM on the system is likewise no problem if you have a winchester or use a RAM disk program.

So, after all my confused explanation, here's the plan for what we give MS-DOS:

- send "COMMAND.COM" as the PATH, send the entire command line as the COMMAND, don't bother with the two FCB pointers, and send 0000 as the environment pointer (this says "use parent's environment").

Before we get all hot headed and spit out the code though, there are a couple details that need recognition:

1. The PATH string is a "C-style" 0 terminated string, not a Pascal string.
2. The COMMAND string is a Pascal string with an added Carriage Return (^M) at the end. We'll have to add the 0 and the ^M to the ends of the strings and send the address of PATH[1] instead of just PATH.

And Finally The Code

See Figure 1 for the Pascal procedure (actually it's a FUNCTION, so it can return an error code). It's a general purpose Exec for MS-DOS. I've been using this function for about six months now and I really can't imagine living without it. Things like menus and shell escapes just seem to flow off...
the fingertips now that I have this tool. See Figure 2 for an example program that uses the Exec procedure.

The way Exec has been written, it will execute any single command it receives. If you send it a null string ("") it will "escape" temporarily to a new copy of COMMAND.COM that you can use until you give the "exit" command. A definite whistle for any program.

A Final Detail

One thing that hasn’t been mentioned until now is Exec's need for "free" memory. What this means is that when Exec is called, there must be a sufficient amount of memory in the machine that has not been allocated to the currently running program. It just happens that if you don’t tell Turbo any different, it will keep ALL of the available memory for itself and you’ll get a big fat error when you try an Exec. This problem is easily solved with the m <A>ximum free memory setting in the compiler <O>ptions menu of Turbo. For most simple programs I can set it down to about 800, but you may have to put it at 1000 or higher for some large applications. But it should be at least low enough that it leaves enough memory for any programs you’ll call from the program you’re compiling (and for any programs that the programs may call, and ...)

Even if you write programs that don’t use Exec you should set the free memory anyway, just in case you happen to be using the program with some multwindow add-on or other schizo type of program. Politeness Man says: "Take only what you can use. And don’t spit in your napkin again or I’ll slap you."

Second Verse, Same As The First

Since I do at least as much programming in Assembly as in Pascal, I also have the same little trick sitting around for Assembly language. Most all of the same ideas and thoughts hold for the Assembly language version as the Pascal, just a bit of difference in the way of making sure enough memory is allocated.

(continued next page)
Calling Exec In Assembly

The method for calling Exec in Assembly will be to put the address of the command line into registers DS:SI and call Exec:

```
MOV SI,offset COMMAND
CALL EXEC
```

On return, the error code will be in the AX register. Features and internal operation are otherwise exactly the same as the Pascal version.

Exec for Assembly language is in Figure 3. One little uniqueness of the code is that it can be linked with full fledged EXE files as well as COM files. This is possible because I put all the local storage for Exec in the CODE segment, save DS (data segment pointer) on entry, move CS (code segment pointer) to DS while Exec is active, and restore DS before exiting. I did this mainly because I like to make COM files as much as possible; they're much smaller. If you'll be doing only EXE files then you should put the data in a separate segment (called DATA maybe?).

LINK Segments

While I'm on the subject of segment names, I should mention the SEGMENT directives in both Exec and the test program. You'll notice first that both SEGMENT directives are exactly the same; this is the only way you can have both sections of code in the same segment (unless you associate the segments with the GROUP directive). Since Exec is declared as a NEAR procedure, it MUST be in the same segment as the main program. Of course, if your program were really huge, you could change it to a FAR procedure and put it in another segment.

Also note the use of PUBLIC in the SEGMENT directive. If you don't declare the two CODE segments as PUBLIC, the code of Exec will be assembled with origin 0 (since it has no ORG statement). This is no problem for the code since 8086 code is all totally relocatable (except FAR jumps and calls). The problem is the data in the segment; all references to the data addresses will be wrong! (I.e., references to code are all relative, but direct references to data are absolute within the current segment.) I spent about an hour on this one (why am I executing the bottom of the base page instead of COMMAND.COM?), but there's no reason you should, so just use PUBLIC and shut up. Okay????

Taking Care Of Memory

If you're assembling to a COM file you'll have to add the following lines to the beginning of your program to free up extra memory:

```
MOV BX,xxxxh
MOV AH,4Ah
INT 21h
```

where xxxx is the total number of paragraphs of memory your program will use for data and code (a paragraph is 16 bytes). See the example in Figure 4.

If you're assembling to an EXE file then you can't release memory already allocated to your program; you'll have to make sure it's never allocated. Do this by using the /c option when you're linking something like the following:

```
LINK /c:1 program+ASMEXEC
```

The /C:1 option keeps one paragraph of extra memory BEYOND declared data for the calling program and frees the rest. For your program you may need to increase the number of paragraphs.

(Note: the /c option is only on Microsoft LINK v3.0 and above. It DOES NOT WORK on the old version 1.0 LINK included with MS-DOS 2.11 and older versions of MASM.)

And Yet Another Exec

While I'm showing Execs for different languages I may as well show one for (shudder) Microsoft Compiled BASIC. Really, I swear I didn't write this one for myself. It was done because of a "request turned to dare." The only program I've used it with is the test program in Figure 5.

(continued on page 46)
HARD DRIVES

LIMITED SUPPLY ON HAND

<table>
<thead>
<tr>
<th>Name</th>
<th>Model</th>
<th>Megas</th>
<th>Size:</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tandon (New)</td>
<td>262</td>
<td>26</td>
<td>5/8 or 3/4</td>
<td>$399.95</td>
</tr>
<tr>
<td>Rodine (New)</td>
<td>202E</td>
<td>26</td>
<td>5.25 Full</td>
<td>$399.95</td>
</tr>
<tr>
<td>CMI (Used Tested)</td>
<td>6426</td>
<td>26</td>
<td>5.25 Full</td>
<td>$399.95</td>
</tr>
<tr>
<td>CMI (Used Tested)</td>
<td>6640</td>
<td>40</td>
<td>5.25 Full</td>
<td>$489.95</td>
</tr>
<tr>
<td>Quantum (New)</td>
<td>Q540</td>
<td>45</td>
<td>5.25 Full</td>
<td>$699.95</td>
</tr>
<tr>
<td>Atasi (Used Tested)</td>
<td>3046</td>
<td>46</td>
<td>5.25 Full</td>
<td>$699.95</td>
</tr>
<tr>
<td>Maxtor (Used Tested)</td>
<td>1050</td>
<td>50</td>
<td>5.25 Full</td>
<td>$649.95</td>
</tr>
<tr>
<td>Micropolis (New)</td>
<td>1325</td>
<td>85</td>
<td>5.25 Full</td>
<td>$1,194.95</td>
</tr>
<tr>
<td>Seagate ST419</td>
<td>15</td>
<td>5.25 Full</td>
<td>$359.95</td>
<td></td>
</tr>
</tbody>
</table>

CONTROLLER CARDS

D.T.C. #03-D (shown with optional software and adapter) control two 8" 80-DOS and two 8" 1000 type hard drives.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$249.95</td>
<td></td>
</tr>
</tbody>
</table>

D.T.C. $08-B control four 8-3/4" DDSO and two 3-1/2" type hard disks.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$69.95</td>
<td></td>
</tr>
</tbody>
</table>

HOST ADAPTER FOR ABOVE CONTROLLERS (for most popular Z-80 computers)
The Vivid Adaptor and software

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$59.95</td>
<td></td>
</tr>
</tbody>
</table>

Xerox Controller; auto boot with your software (advanced diagnostic disk) for XT PC clones with up to 33 meg.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$119.95</td>
<td></td>
</tr>
</tbody>
</table>

SA 450 FLOPPY DRIVES

DDS (Brand new in boxes) 48 T.P.I.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Height: 5.25</td>
<td>$49.95</td>
</tr>
</tbody>
</table>

THE SENSIBLE 5 1/4" DISK DRIVE CABINET KIT

- EMI Filter, 6 AMP
- Surge Protection
- Power Switch
- Power Supply Shield
- Connector and Fan Covers

Constructed of heavy gauge aluminum, this utility disk drive case will accommodate: 5 1/4" hard disk, floppy drives, half height, full height, 3/4 height, controller card and single board computer. Overall dimensions: 10" x 8" x 7"

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ONLY $39.95</td>
<td></td>
</tr>
</tbody>
</table>

SALE $69.95 Without Power Supply

GO ON LINE WITH UNITED PRODUCTS

Take advantage of the many items that space does not allow us to advertise. Use your computer to down load our weekly specials and hard to find items.

Call us 24 hours a day, 7 days a week: 300 or 1200 baud, even parity, 8 bits with one stop bit.

The phone number is: (206) 483-2583.

New items are added weekly. Don't miss out, call each week.

CONTROLLER CARDS

D.T.C. #03-D (shown with optional software and adapter) control two 8" 80-DOS and two 8" 1000 type hard drives.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$249.95</td>
<td></td>
</tr>
</tbody>
</table>

D.T.C. $08-B control four 8-3/4" DDSO and two 3-1/2" type hard disks.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$69.95</td>
<td></td>
</tr>
</tbody>
</table>

HOST ADAPTER FOR ABOVE CONTROLLERS (for most popular Z-80 computers)
The Vivid Adaptor and software

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$59.95</td>
<td></td>
</tr>
</tbody>
</table>

Xerox Controller; auto boot with your software (advanced diagnostic disk) for XT PC clones with up to 33 meg.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$119.95</td>
<td></td>
</tr>
</tbody>
</table>

SA 450 FLOPPY DRIVES

DDS (Brand new in boxes) 48 T.P.I.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Height: 5.25</td>
<td>$49.95</td>
</tr>
</tbody>
</table>

THE SENSIBLE 5 1/4" DISK DRIVE CABINET KIT

- EMI Filter, 6 AMP
- Surge Protection
- Power Switch
- Power Supply Shield
- Connector and Fan Covers

Constructed of heavy gauge aluminum, this utility disk drive case will accommodate: 5 1/4" hard disk, floppy drives, half height, full height, 3/4 height, controller card and single board computer. Overall dimensions: 10" x 8" x 7"

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ONLY $39.95</td>
<td></td>
</tr>
</tbody>
</table>

SALE $69.95 Without Power Supply

GO ON LINE WITH UNITED PRODUCTS

Take advantage of the many items that space does not allow us to advertise. Use your computer to down load our weekly specials and hard to find items.

Call us 24 hours a day, 7 days a week: 300 or 1200 baud, even parity, 8 bits with one stop bit.

The phone number is: (206) 483-2583.

New items are added weekly. Don't miss out, call each week.

SWITCHING POWER SUPPLY

Real Power from a Famous Manufacturer delivers 10 amps at 5V, 8 amps at 12V measures only 8 1/2" by 4 1/2" and only 2 1/2" high.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDK model #REX-100T-1</td>
<td>$29.95</td>
</tr>
</tbody>
</table>

with prices

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$39.95</td>
<td></td>
</tr>
</tbody>
</table>

BUILDING A ROBOT? WE CAN HELP!

STEPPER MOTORS

Superior model #PM4-FD03 1/8" shaft, 200 steps per rev., 3/16" shaft, 300 steps per rev., 1.5V of torque 1V in 1 amp

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$9.95</td>
<td></td>
</tr>
</tbody>
</table>

Copl model #SP-S7 1/8" shaft, 7.5 steps 1/2V, 36 OHM

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$6.95</td>
<td></td>
</tr>
</tbody>
</table>

UNIVERSAL STEPPER MOTOR DRIVER KITS

FOR DIRECT CONTROL FROM YOUR COMPUTER

- Control almost any stepper motor, forward/REVERSE and step sequence.
- For stepper motors rated 12.0 VDC.
- Uses a power supply with an output between 3.9 VDC and 12.0 VDC.
- Power supplies must be able to provide 200 ma for a motor rated above 12.0 VDC.
- Inputs/outputs are on a standard 22 pin edge card connector (54 spacing) 1/4 x 1/6 card, 20 x 10 size.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>TECHNIK-002 /USM-D/C</td>
<td>$29.95</td>
</tr>
</tbody>
</table>

SAME SPEC'AS USM-D/C except a stand-alone driver
- On board clock (user optional)
- No edge connector

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>TECHNIK-002 /USM-D/C</td>
<td>$199.95</td>
</tr>
</tbody>
</table>

POWER SUPPLY KITS

(suitable for above)

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$69.95</td>
<td></td>
</tr>
</tbody>
</table>

(suitable for above)

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$69.95</td>
<td></td>
</tr>
</tbody>
</table>

MONITORS

(BARE FRAME)

Bull Brothers TV-120 (shown)

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$49.95</td>
<td></td>
</tr>
</tbody>
</table>

Audiontronic 964-04

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$49.95</td>
<td></td>
</tr>
</tbody>
</table>

TTU/XY Xerox 820 compatible

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$39.95</td>
<td></td>
</tr>
</tbody>
</table>

TTU/XY Xerox 820 compatible

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$49.95</td>
<td></td>
</tr>
</tbody>
</table>

NEC MD AN8

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$49.95</td>
<td></td>
</tr>
</tbody>
</table>

TTU/XY Xerox 820 compatible

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$39.95</td>
<td></td>
</tr>
</tbody>
</table>

TTU/XY Xerox 820 compatible

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$49.95</td>
<td></td>
</tr>
</tbody>
</table>

Audiontronic 918-14

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$49.95</td>
<td></td>
</tr>
</tbody>
</table>

Composite video or

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$49.95</td>
<td></td>
</tr>
</tbody>
</table>

TTU/XY

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$49.95</td>
<td></td>
</tr>
</tbody>
</table>

BUSINESS TELEPHONES

BRAND NEW 6 button business telephones all the standard features including buzzers and ringer locks the same as picture. Color, choc. brown.

CLOSE OUT

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$49.95</td>
<td></td>
</tr>
</tbody>
</table>

FLUKE MULTIMETERS

Model #23

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$135.99</td>
<td></td>
</tr>
</tbody>
</table>

Model #77

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$129.95</td>
<td></td>
</tr>
</tbody>
</table>

DC FANS

12 Volts 6" 5V, 4.5"

<table>
<thead>
<tr>
<th>Name</th>
<th>Price:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$19.95</td>
<td></td>
</tr>
</tbody>
</table>
If you really want to know about the internals of calling Assembly language from BASIC then I suggest you read the manual because I don’t have the patience to go through that hell again. Just be warned that the BASIC manual not only has typos, it has blatant, obnoxious misrepresentations too. Otherwise see Figure 6 for the BASIC version of Exec.

To use the BASIC version, you must assemble it with MASM and then link it with your BASIC object module with the following LINK statement:

```
LINK/c:1800 program+BASEEXEC
```

Don’t ask ME why BASIC needs 92K (1800h paragraphs) of extra free memory. It just DOES.

Other high level languages (especially Microsoft’s) should use routines similar to BASIC’s (i.e., passing the parameters on the stack), although the reasons behind their respective parameter passing methods will probably be less obscure.

Enough Is Enough Is Enough

Well, TRT (Turkish Radio and Television) is finished with the traditional music now and going on to European avant-garde jazz so I guess that’s as good an excuse as any to stop shooting off my mouth and finish this up. Hopefully somebody somewhere will get some use out of my little contributions, otherwise at least I can be happy knowing that I have.

As DEVO once said: “Take this, brother, and may it serve you well.”

Next Time: “Home, Home on the Range” and other topics in environmental awareness. See you at SOG.

---

**Figure 6 - BASIC Version Of EXEC**

```
;----- BASEEXEC - execute command line in string
; pointed to by stack
;
; This routine is expected to interface with
; Microsoft Compiled BASIC
; calling conventions
;
; Currently it is assumed that the system has
; a drive C available (usually a winchester) that
; contains COMMAND.COM in the \ directory.
;
; CALL FROM BASIC:
; CALL EXTEXEC(CommandString,ErrCode)
;
; where CommandString is a string & ErrCode is INTEGER;
;
; On return, ErrCode will be 0 if everything went ok,
; otherwise it will correspond to the error codes
; documented in the MSDOS
; programmer's manual under function call 4Bh
;
; LINK PROCEDURE:
; LINK/c:1800 program+BASEEXEC
;
; On entry to EXTEXEC, immediately after pushing BP,
; the stack looks like the following:
;
; +--------------------------+<--- SP+8 <--- BP+8
; + ptr to CommandString + <--- SP+6 <--- BP+6
; +--------------------------+<--- SP+4 <--- BP+4
; + Return Segment + <--- SP+2 <--- BP+2
; +--------------------------+<--- SP <--- BP
;
; CONST segment byte public 'CONST'
; CONST ends

DATA segment word public 'DATA'
DATA ends

DGROUP group DATA,CONST

CODE segment byte public 'CODE'
public EXTEXEC
assume cs:code, ds:dgroup

EXTEXEC PROC FAR
JMP short EXTEXEC1

PATH DB 'c:\command.com',0
COMMAND DB 0,'/c '
DB 60 dup (?)
```
PARMS DW 0,0,0,0,0,0,0
comofs equ word ptr 2
comseg equ word ptr 4

EXTEXEC1:
    PUSH BP
    MOV BP, SP
    PUSH BP
    MOV BP, SP
    MOV ES, CX
    MOV DI, offset COMMAND
    MOV CS:BYTE PTR [DI], 0
    MOV SI, [BP]
    MOV [SI], '1'
    MOV CX, [SI]
    MOV SI, [SI+2]
    JCXZ EXTEXEC2
    MOV CS:BYTE PTR [DI], 0
    ADD DI, 3
    MOV AL, CL
    ADD AL, 3
    STOSB
    ADD DI, 3
    REF MOVSB
    MOV AX, 0000h
    STOSB

EXTEXEC2:
    PUSH DS
    MOV CX, CS
    MOV DS, CX
    MOV AX, 4B00h
    INT 21h
    POP DS
    JC EXTEXEC9
    MOV AH, 4Dh
    INT 21h
    XOR AX, AX
    POP AX
    POP AX
    POP BP
    RET 4

EXTEXEC9:
    MOV SI, [BP]
    MOV [SI], AX
    XOR AX, AX
    POP AX
    POP ES
    POP BP
    RET

EXTEXEC ENDP
CODE ends

CPM-80 LIVES ON YOUR PC
CP/Mulator puts a 4mhz 8 bit CPM-80 emulator in your IBM-PC for only $99.

- Execute 8 bit CP/M programs at 4 mhz. Stop waiting around for those slow software emulators at twice the price of this speed demon!

- CP/Mulator makes a fantastic CP/M development environment and allows you to use all your CPM-80 tools, and the IBM-PC tools too!

- Why write off that expensive package that you bought for CPM-80 just because you upgrade to a IBM-PC? Apply the $99 solution and keep it working. No retraining or conversion blues!

- CP/Mulator includes the NEC uPD70108 processor which increases the speed of your IBM-PC by at least 10%. 100% compatible with existing IBM-PC software and 8087 compatible.

PURCHASE THE SOFTWARE ALONE FOR ONLY $69!!
Source Information
P.O. Box 2974
Warminster, PA 18974
Phone (215) 441-8178

---

HiSpeed C Library
The HiSpeed replacement library for CP/M Toolworks C/80 uses either Microsoft Macro-80/Link-80 or SLR Systems SLRMAC/SLRNX.

Compile HELLO.C into a 254-byte HELLO.COM in 10 seconds with a single command line.

Features:
- Fast K&R incremental development.
- Full STDIO stream library: fseek, fwrite, etc.
- Function calls match major MS-DOS, CP/M-68K and UNIX compilers. Painless portable code.
- Full math, strings, searching, sorting, timing.
- One function per page documentation with examples that work (410 pages, 310+ functions).
- Utilities to archive, bold-print, beautify, cross-reference, function map, ... (27 in all).

Printed reference manual and object code
$29.95

One megabyte of source code on disk
$19.95

Free brochure. Phone orders shipped immediately. All Media Master formats and 8” ssdd.

Viking C Systems
2243 South Belaire Drive
Salt Lake City, UT 84109
1-801-466-6820
ask for Grant
Clipper, A Fast Solution To dBASE

A database lies at the heart of nearly every business related application, and for years the most popular database language has been dBASE II. So, it's not unusual to run into someone who wants a dBASE II (or dBASE III) system upgraded. I was faced with such a project recently and I learned a lot.

Two weeks before copy deadline for issue #30 the old BBII in the mailroom started showing signs of senility. That's the system we use for our mailing list (it's the only CP/M machine with a large enough winnie), so its health is critical.

I can't say it didn't give us some warning. Six months ago the BBII suddenly refused to run any submit programs (EX14, Supersub...). That was irritating (and we never figured out why) but we learned to type programs (EX14, Supersub...).

Two weeks ago it died right in the middle of data entry. Cary paged me back up the files on the winnie - so its health is critical. I spent an hour or so finding out which files had been corrupted (hadn't been closed) and another four or five hours recovering as much as I could. (Cary would reenter the rest.)

After it quit two days in a row (we've never had a convenient way to back up the files on the winnie - which meant we lived dangerously), I decided I had to make the leap. I had to move all the dBASE II files onto a clone and run them under something. Hopefully that something would be faster and more dependable than dBASE II.

Clipper

At Comdex, Sandy saw a new version of Champion (a gigantic accounting package written in dBASE). It turned out that the new Champion wasn't running under dBASE any longer. Oh, the source code was pretty much the same, but it had been compiled by Clipper. The difference in performance was impressive.

Well, knowing how fussy these guys are about their program, and how much effort they had put into making dBASE work (wrote their own run-time package for dBASE II), I figured that if they liked Clipper then I probably ought to take a look at it.

So, a few months ago I got ahold of the Clipper folks and asked for a review copy. They sent version 1.0. (It turned out that the timing was great.)

Back At The Office

So I was faced with a dying BBII and an untried Clipper compiler. I could kill two birds with one stone, but I had to do it quickly. Orders weren't getting out because Cary couldn't make receipts, and Becky was trotting about the office clutching her calendar lest someone (I) might forget that May 1 was a typesetting deadline.

I started futzing over the files, and since I hadn't purchased dBASE III I knew I was facing a challenge. (dBASE III comes with a routine that translates dBASE II source into III.) Clipper only understands dBASE III syntax.

A trip to the bookstore turned up "Understanding and Using dBASE II & III" by Rob Krumm. I figured any book that dealt with both languages side by side would help me translate my code over from II to III. Boy, was I wrong. Within hours I was back at the bookstore.

I looked more closely this time. After an hour of careful digging I found a really good book — "dBASE III For The Programmer" by Nelson Dinerstein. (The book store was nice enough to take back Krumm's book. Hooray for Waldenbooks.)

Developing Code

The Clipper compiler spits out some pretty good error messages, which helps a lot when it's just you and a book and 100K of dBII code. A compiler isn't the fastest way to develop code, but with Becky and Cary (and all those orders) waiting, it didn't take too long. Old Faithful (my Challenger 186 board) definitely earned its keep during this project.

A Clipper batch file took care of the compilation and linking. So when I was ready, I'd just fire up the .bat file and take a 5 minute break. When it finished I had a standalone .EXE file.

One thing I always look at when trying out a new compiler is its overhead. How big a .COM or .EXE file do you start with before you can do something useful?

So I wrote a one-liner (program).

? "Hello World"

Now any self-respecting C compiler would recognize that immediately (I suspect that some C compilers are optimized for "Hello World") and generate an 8K .COM file. Clipper generated a 115K .EXE file. The compilation took 1 min. 31 sec. on Old Faithful. (The compiler reported 17 bytes of code, the rest was library.)

My database program is made up of 38 dBASE II (now dBASE III) files totaling a little over 100K of source. Total time to compile and link takes 5 min. 23 sec. Clipper generates a single 157K .EXE file. I can live with that since the finished product runs very, very fast.

How Fast Is It?

It's so fast that screens come and go instantly (while the same code running on a borrowed copy of dBASE III+ was very, very slow, significantly slower on the 6MHz clone than dBASE II was on the 4MHz Z80.) During data entry, file updates that take 15 or 20 seconds on the Z80 don't even break a typist's stride.

I wrote a program made up of three giant case statements. Its task was to save Cary six hours of hand counting.
We have to tell the post office how many subscribers we have in each zone (for each issue). There are eight zones (and about a dozen different ZIP code ranges for each zone). Plus, if a copy goes to an advertiser or my mother, then it must be counted differently. (Mothers don’t realize what a pain they are — almost as bad as advertisers. But it’s hard to do without them.)

Anyway, the Clipper-compiled program checked all 13,000+ entries in our database and printed out the totals in 12 minutes 0 seconds.

I then ran dBASE III+ on the same source code. After 1 hour and 20 minutes, I interrupted it to see how it was doing. It had checked 1302 records. At that rate it was going to take dBASE III over 13 hours to finish. Both times were taken on the same 6MHz, sped-up clone.

This trial, in fact, points out the strengths of both Clipper and dBASE III.

dBASE III is interruptable. You can check to see if something strange is happening by looking at the contents of all the variables. Plus it’s an interpreter, so you’re running again as soon as you make a change. No compiler to get in the way. These features make dBASE III a natural for software development. dBASE also lets you BROWSE through the database, making corrections or additions.

However, dBASE III is slow. Very, very slow. It would be slow on an AT; I don’t know how people stand it on a PC or XT. Clipper, on the other hand, is very fast. In fact, you’d have trouble generating a faster .EXE file no matter what language you chose. Plus, Clipper doesn’t charge you a fee every time you distribute a program. You just compile and ship.

General Feelings
I really like Clipper. I’m not sure I would have purchased it for $695 under normal circumstances, but considering the fix I was in and considering Champion’s recommendation, I would have.

The speed difference between it and dBASE is astounding. You have to see it to believe it. Another plus — it seems to have few of the undocumented peculiarities that have been a trade-mark of dBASE. No question, this is a professional package.

However
Clipper is copy protected. Boo. The protection scheme is relatively easy to live with (it lets you put up to four copies onto any mix of floppy and winchesters, and it lets you uninstall any of those copies and move them).

Clipper also has at least one bug. A number of times, I had the 13th line of the display show up at the top of the screen. That was during the development process when I was also finding undefined variables or other strangenesses in my code. I haven’t had the problem lately.

With any compiler, you lose the immediate mode of dBASE when you compile. BROWSE is gone; so is “display memory.” These are nice touch-es, and important, especially during development.

Support
Nantucket definitely had paper saving in mind when they wrote the manual. It’s the shortest thing I’ve seen since Cheerios sent Erin a free coloring book. There are almost no examples of how to use the compiler and linker directives, and I found the text very obscure. Fortunately, most of the time you’ll be able to live with the defaults (still, an example, or a contact person who’d take a quick phone call would have saved me about 10 hours).

I also tried to use the built-in debugger. It offered a way of tracing through the code step by step, watching variables change, or triggering on

(continued next page)
a breakpoint. Great, I thought, and turned to the chapter on using the debugger. The chapter turned out to be four partially-filled pages. I spent a whole day fighting the debugger. I still don’t know how to make it do the fancy stuff.

When the 13th line problem started cropping up, I got serious about finding support. Page A1 of the manual states:

"Nantucket provides two sources of technical support for registered users.

1. On-line assistance via The Source.

2. Direct assistance from Nantucket. There is an additional charge for this service. A Mastercard or VISA is required."

Okay, I’m not a member of The Source (we don’t have a phone access node in Bend for any commercial network), and I had no desire to join and pay monthly fees just to ask Clipper a question about its compiler.

So, VISA card in hand (I try my best to review like a user, not like a reviewer), I picked up the phone. Unfortunately there was no phone number on the support page. In fact, I didn’t find a single phone number in the entire manual. (No address, either.)

I got the hint. (And things turned out pretty well.)

General Notes

1. Take your time futzing over dBASE II files. Get a good book and use it. The code isn’t all that different, and once you get the hang of it you’ll zip right along. But it has to be done and it takes time.

2. Translate all dBASE II data files into SDF (standard data format) and then reconstruct under Clipper. (dBASE III didn’t translate back from SDF properly, at least not for Clipper.)

3. Clipper creates its own index files. They are not interchangeable with dBASE III (although dBASE has no trouble reading Clipper data files).

4. Networking is the big thing now for businesses. Any database package running on a network must support record locking so that several people can safely access the same database at the same time. According to the May 19 issue of InfoWorld, Lanlock (by Delta Contracting Services, whoever they are) is a $100 package that upgrades Clipper to support record locking for LAN use. Nantucket is supposed to be working on a similar add-on for Clipper. dBASE III+ is supposed to have an expensive upgrade which does record locking, but users are reporting that it doesn’t work.

Finally

I didn’t set out to look for nooks and crannies in Clipper. I used it to compile a major database program. Once I had made all the dBASE II to dBASE III translations, it worked.

I usually don’t consider packages that are this expensive. After all, Turbo Pascal is more powerful than dBASE and much less expensive. There are also database packages for a lot less. But if you’re familiar with this language and you need to get something out quickly, Clipper is not a bad way to do it. (Especially if you’re already a member of The Source.)

Clipper
Nantucket, Inc
$695
5995 Sepulveda Blvd
Culver City CA 90230

Advanced Concepts Engineering & Consulting
8526 S.W. 17th Street
Boca Raton, Florida 33433

Advaced Concepts “Mini-Winnie”

HARD DISK SYSTEMS

for your Z-80 Computer!

Our EXTERNAL systems install in seconds! Includes an attractive vertical cabinet with integral P/S and fan, 3 foot cables. Everything necessary to plug in and go!

Our INTERNAL systems do not displace a floppy drive, do not require a power supply upgrade, use the highest technology winchesters available. No compromises.

COMPLETE SUBSYSTEMS

Plug in and run includes software, documents, cables, brackets as req’d.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5mb External</td>
<td>10mb External</td>
<td>20mb External</td>
<td>10mb Internal</td>
<td>20mb Internal</td>
</tr>
<tr>
<td>Kaypro (1,2,4,10)</td>
<td>Big Board II</td>
<td>Xerox 820-II</td>
<td>Micro Mini SB-180</td>
<td>Gemini Zorba</td>
</tr>
<tr>
<td>Osborne</td>
<td>Sanyo MBC-1150/1160/ual</td>
<td>AVL Envouge (with mods)</td>
<td>Epson QX-10</td>
<td>Morrow MD-2.3</td>
</tr>
<tr>
<td>(You may delete $90 for external systems less P/S enclosure)</td>
<td></td>
<td></td>
<td>Add $6.00 for extra long cables.)</td>
<td></td>
</tr>
</tbody>
</table>

You may purchase our controller, software, cables & P/S enclosure for $395. Add your ST-506 family disk drive (up to 26mb) to complete your system!

Compatible with most other add-ons, including Microsphere & Advent RAM disks, SWP CoPower/RAM disk, Pho-5 Family, Advent Turbo ROM, all speedup and double/quad density upgrades, ZCR93, Microshell, Plu Plus Perfect 2.2E.

The key to the Mini-Winnie system is the INSTALL program, which links your drive and your computer, the way YOU want it. Extensive menu options allow selection of any ST-506 compatible drive, as 1 to 4 logical drives within CP/M at any drive letter. Diagnostic and format capability provided to match the installation. Distributed with SOURCE files. No programming required.

We have closed user group. OEM/Dealer programs. Call for quantity schedule. If for any reason you are not satisfied, return for full refund within 30 days. All systems warranted for 90 days. Info and manual $8.00 credited with your system purchase.

ORDER INFO: Call (305) 482-7302. Terms: UPS cash COD, check or M.O. Please allow 4 weeks for delivery. Ria. add 5% tax. Shipping extra. Please specify desired floppy format.

Advanced Concepts Engineering & Consulting
8526 S.W. 17th Street
Boca Raton, Florida 33433
I

Up To Your Ears In Alligators?

If that sounds familiar, you need Write-Hand-Man™, the multi-
function pop-up desktop organizer that works
neatly with existing soft-
ware for CP/M™ 2.2 and
3.0 systems. Write-Hand-
Man eliminates that
swamped feeling with
tools that will get you
organized. Write-Hand-
Man comes with a
4-function, floating-point,
14 digit Calculator – Notepad
– Two-week Appointment
Book, File and Directory viewing – Phonebook
with dialing – Cut and Paste – Key Redefinition –
ASCII II table. Even add your own applications.

Clear the swamp from your desktop.
Order Write-Hand-Man today. $49.95
CA residents add 6.5% tax. Sorry, no
credit cards or purchase orders.
Specify: 8" or which 5" format
CP/M 2.2 or 3.0 format
30 day guarantee
**Write-Hand-Man – Poor Person
Software
**CP/M – Digital Research

TOTAL CONTROL
with LMI FORTH™

For Programming Professionals:
an expanding family of
compatible, high-performance,
Forth-83 Standard compilers
for microcomputers

For Development:
Interactive Forth-83 Interpreter/Compilers
• 16-bit and 32-bit implementations
• Full screen editor and assembler
• Uses standard operating system files
• 400 page manual written in plain English
• Options include software floating point, arithmetic
coprocessor support, symbolic debugger, native code
compilers, and graphics support

For Applications: Forth-83 Metacomplier
• Unique table-driven multi-pass Forth compiler
• Compiles compact ROMable or disk-based applications
• Excellent error handling
• Produces headerless code, compiles from intermediate
states, and performs conditional compilation
• Cross-compiles to 8080, Z-80, 8088, 68000, 6502, 8051,
8096, 1802, and 6303
• No license fee or royalty for compiled applications

For Speed: CForth Application Compiler
• Translates “high-level” Forth into in-line, optimized
machine code
• Can generate ROMable code

Support Services for registered users:
• Technical Assistance Hotline
• Periodic newsletters and low-cost updates
• Bulletin Board System

Call or write for detailed product information
and prices. Consulting and Educational Services
available by special arrangement.

Laboratory Microsystems Incorporated
Post Office Box 10430, Marina del Rey, CA 90295
Phone credit card orders to: (213) 306-7412

Overseas Distributors:
Germany: Forth-Systeme Angelika Flesch, Tittsee-Neustadt, 7651-1665
UK: System Science Ltd., London, 01-248 0982
France: Micro-Sigma S.A.R.L., Paris, (1) 42.65.95.18
Japan: Southern Pacific Ltd., Yokohama, 045-314-9514

Forth-Systeme Angelika Flesch, Tittsee-Neustadt, 7651-1665
UK: System Science Ltd., London, 01-248 0982
France: Micro-Sigma S.A.R.L., Paris, (1) 42.65.95.18
Japan: Southern Pacific Ltd., Yokohama, 045-314-9514

Forth-Systeme Angelika Flesch, Tittsee-Neustadt, 7651-1665
UK: System Science Ltd., London, 01-248 0982
France: Micro-Sigma S.A.R.L., Paris, (1) 42.65.95.18
Japan: Southern Pacific Ltd., Yokohama, 045-314-9514
Allocating Static Arrays At Run-time

If CP/M Turbo Pascal has a main weakness, it's the way it allocates memory at compile time. This is a second discussion of ways around the problem.

In the August '85 issue of Micro C, James Shiflett described how Turbo Pascal can dynamically use all of a CP/M system's memory by manipulating the run-time stack and heap pointers.

This is a fairly simple solution to the problem created when the Turbo compiler locates variables below the compilation ending address. This address must be set wastefully low to insure that the program will run on all systems regardless of size (system size varies from 59 1/2K to 64K on Kaypros alone).

(EDITOR'S NOTE: Turbo puts its variables at the very top of program memory. This causes a problem if you try to run a compiled Turbo program on a system with less program space. The classic way to get around this problem has been to explicitly set the size of the program area to, say, 48K, but in doing this you give up use of a substantial portion of memory.

However, Mr. Shiflett's approach can be used only with dynamic variables and forces the programmer to use pointers, which can unnecessarily complicate some programs.

An Array To Free Memory

Another approach would be to find a way to associate static variables (typically arrays) with whatever free memory is available to the program at run-time. A simple way to do this is as follows:

Insert the following declaration at the top of the program:

```pascal
VAR
  TopOfData : BYTE;
  BDOSAddr : INTEGER ABSOLUTE $0000;
  InitialIndex : INTEGER;
  MaxIndex : INTEGER;
  BytesNeeded : INTEGER;
```

Then, at the beginning of the main part of the program, insert something like this:

```pascal
BEGIN
  Assign or compute BytesNeeded here.
  InitialIndex := ADDR(TopOfData) + 256;
  MaxIndex := BDOSAddr - 1;
  IF (MaxIndex - InitialIndex + 1) < BytesNeeded THEN
    BEGIN
      WRITELN('Not enough memory');
      HALT;
    END;
  (Call main menu here, for example)
END.
```

Now compile the program to a .COM file, using any reasonably small ending address. At run-time, any space left between the top of the program and the operating system will then be available to the program as an array of type BYTE which can be referenced using Turbo's built-in MEM array. Any portion of this array beginning with element MEM[InitialIndex] and ending with MEM[MaxIndex] can then be used freely by the program.

If an array of some other type than BYTE is needed, for example an array of type INTEGER, then add to the above declarations as follows:

```pascal
VAR
  AnArray : ARRAY [1..32767] OF INTEGER ABSOLUTE $0000;
```

Or, for a REAL array, use this declaration:

```pascal
VAR
  AnArray : ARRAY[1..10922] OF REAL ABSOLUTE $0000;
```

Now, in the main body of the program, compute the initial and maximum allowable indices for the array:

```pascal
InitialIndex := (ADDR(TopOfData) + 256) DIV SIZEOF(AnArray[1]) + 1;
MaxIndex := ((BDOSAddr - 1) DIV SIZEOF(AnArray[1])) - 1;
```

This array can then be used just like any array as long as you stick to those elements between InitialIndex and MaxIndex.

(Note: TopOfData is not quite the same as the program's ending address, hence the use of 256 to make sure we're past the ending address. The exact number of bytes between the top variable and the ending address is 137 with Turbo version 3.0, but 256 can be used to be safe.)

Of Course There's A Risk

One disadvantage to these two approaches is the sacrifice of run-time index range checks. If a program has a bug in it which results in the use of an index value outside the range InitialIndex..MaxIndex and the program doesn't check for this, it could write over part of itself or the operating system in memory. Of course, this is also a risk in using pointers to dynamic variables.

And A Solution

A third approach, which retains run-time range checking below the array's smallest index and thereby reduces the chance of a program bug writing over part of the program, is as follows:

First compile the program as suggested in Mr. Shiflett's article to find the minimum ending address. Then declare a global array absolute to that address plus one:

```pascal
CONST
  EndingAddrPlus1 = {ending address + 1 goes here}
VAR
  AnArray : ARRAY[1..20000] OF INTEGER ABSOLUTE EndingAddrPlus1;
```

The beginning and ending array indices would then be:

```pascal
InitialIndex := 1;
MaxIndex := ((BDOS - EndingAddrPlus1) DIV SIZEOF(AnArray[1])) - 1;
```

A program bug causing a computed array index to drop below InitialIndex would generate a run-time error when attempting to reference the array, thereby preventing the program in memory from being corrupted and causing unpredictable results.

By Philip Hess

7917 N 10th Street
Lafayette IN 47904
EXPAND YOUR KAYPRO OR XEROX

Most versatile. Easiest to use. Finest quality. Claims are one thing, but the proof is in our products. We guarantee satisfaction.

Nothing can improve your computer's productivity like a hard disk.

HARD DISK SYSTEMS by Emerald Microwave

If you have been thinking about moving up to a hard drive, why wait any longer? We have everything you need from the host board to a complete turn-key system. Runs on the Xerox, Kaypro, or almost any 280 system.

HDS Host Board — The hardware you need to connect your computer to your hard disk controller board.
- Interfaces to the WD1002 controller board
- Plugs into the 280 socket, no other wiring required
- Switch selectable I/O addressing
- Comes fully assembled and tested

THE WINCHESTER CONNECTION by MICROCode Consulting

The most comprehensive hard disk software package that you can buy.
- Works with one or two hard disks — 5 to 64 meg
- Menu operate install, no software to assemble
- Complete testing and error handling
- Warm boot from hard drive
- BIOS drivers install above or below CP/M, your option
- Allows custom partitioning and mixed drives types
- Includes manual, format, test, park, backup, and swap utilities

WD1002-05 HARD DISK CONTROLLER BOARD by Western Digital
- 5 ¼” profile with standard power connector (+5 only required)
- 40 pin host interface
- Connects to ST506 compatible drives
- WD2797 floppy disk controller and interface on board
- Can control up to three hard drives and four 5 ¼” floppy drives

COMPLETE HARD DISK SUBSYSTEMS by Emerald Microwave
- Hard disk subsystems with host, cables, software, hard drive, and cabinets with power supply. Fully tested and ready to go.
- Internal hard disk kits available for Kaypro computers
- 10 and 20 meg units in stock, others by order
- Available for Kaypro, Xerox, Morrow, and others

10 Meg Hard Drive Subsystem ........................................ $ 745.00
20 Meg Hard Drive Subsystem ........................................ $ 940.00
Host board with software ............................................. $ 89.00
WD1002-05 Controller Board .......................................... $185.00
20 Meg Kaypro Internal kit w/KayPLUS Rom ...................... $995.00

ACCESSORIES FOR THE KAYPRO AND THE XEROX 820

Xerox 820-2 Main Computer Board .................................. $ 95.00
Xerox 820-2 Floppy Controller board ............................... $ 75.00
Xerox 820-2 Main board w/Floppy Controller ..................... $150.00
Xerox 5 ¼” Drive cabinet w/ribbon .................................. $ 25.00
Xerox internal video cable .............................................. $ 80.00
Board mount power connector ...................................... $ 2.50
Parallel ASCII keyboard (not standard Xerox) ..................... $15.00
Dual 5 ¼” Disk Drives — DDS 48 TPI, in cabinet with standard Xerox cable .................................................. $265.00
Panasonic DDSD 48 TPI Disk Drives................................. $114.00
Panasonic DSDD 96 TPI Disk Drives................................. $129.00
UniForm ................................................................. $ 64.95
UniDOS for IBM ........................................................ $ 64.95

We guarantee satisfaction on all of our products or your money back. Include $4.00 shipping and handling, $7.00 for COD, call for Blue Label charges. Phone hours: 8:30am to 5:00 pm weekdays; check our bulletin board — QP/M, 5:30pm to 8:00am seven days a week, for our latest stock and prices.

WE SUPPORT OUR CUSTOMERS

INTRODUCING THE KayPLUS ROM PACKAGE by MICROCode Consulting

The most important element in the performance of your Kaypro computer is its monitor rom. With KayPLUS you get all the advantages of a Kaypro 10, even on your Kaypro 2.
- Install up to four floppies and two hard drives
- Boots from floppy or hard disk
- Selectable keyboard buffer
- Automatic screen blanking
- 12 disk formats built in
- Full automatic disk relogging
- Internal real-time clock and ram card support
- Can use 96 TPI and 3 ¼" disk drives
- Includes manual, format, configuration, diagnostics, syngon, and hard disk utilities
- No software assembly required
- Available for '83 and '84 series Kaypros — Specify Model

KayPLUS ROM Set ...................................................... $ 69.95
KayPLUS ROM Set with QP/M ........................................ $125.00

Need to keep better track of your files?
Date stamp them with QP/M.

QP/M by MICROCode Consulting

Full CP/M 2.2 compatibility with many more features. You've seen the replacements that eat up memory and need auxiliary programs to run. Not QP/M. Fifteen internal commands, automatic disk relogging (no more control C), user area selection from colon, 31 user areas, drive search path, archive bit maintenance, and transparent timemdate stamping, all in the same space as CP/M 2.2. Installs from a convenient customization menu, without any software to assemble.

QP/M Operating System, complete bootable copy .................. $ 80.00
QP/M without BIOS .................................................... $ 60.00

Still running single density on your 820-1?

PLUS2 ROM by MICROCode Consulting

X120 DOUBLE DENSITY BOARD by Emerald Microwave

Clearly the most versatile double density package for the Xerox 820-1. Run up to four floppy disk drives at once, both 8" and 5 ¼" at the same time. Software compatible with Kaypro and Xerox 820. Supports all standard printers, and most add ons like the Ferguson Ram Board. You get mini-monitor functions, autoboot capability, 19 built in disk formats, and bank mode operation for more space in your TPA. Lets you run 48 TPI disks on 96 track drives. Works with UniForm and QP/M.

PLUS2 ROM Set and X120 Board A&T ................................. $ 135.00
PLUS2 ROM Set and X120 Bare Board .............................. $ 62.00
PLUS2 ROM Set only .................................................. $ 49.95
120 Bare Board only ................................................... $ 22.00
Other kits, parts, and packages available

EMERALD MICROWAVE

P.O. Box 6118 Aloha, OR 97007 (503) 642-1860
The Kaypro 84 Video Circuit: Yet Another Fix

It has to be an awfully dark cloud to be completely devoid of useful information, but my experience with a fussy Kaypro 10 had me wondering if I hadn't found a graphic case of just that kind of cloud. If your 10 or Kaypro 84 is having video problems, scan the following.

Sometimes it seems that Murphy is just waiting for me to get out of bed in the morning (which no doubt explains why I'm a night person). On one such morning, I headed down to MicroSphere to drop off one of our Kaypro 10s.

"Sure, it works fine," I said, setting it carefully on their bench (can't be too gentle with the winnie) and heading off for the Micro C office.

An hour later Don was on the phone.

"Your 10 doesn't display the sign-on menu."

"I never use the sign-on menu — it's just rusty."

There was a long pause. Don doesn't understand computers the way I do.

However, the next day I got another call.

"Your 10 doesn't display the sign-on menu."

All right, already. Don was getting as monotonous as those ingrates who want us to go monthly. I headed over to the MicroSphere office expecting the problem to be video RAM or 6545 video controller or something common like that.

Unfortunately, Don had tried all those things (and had tried three different copies of menu).

Well, it wasn't as bad as it sounded. The menu was there, at least the text part — plus two or three extra characters scattered about the screen for special effect. However, the boxes that were normally drawn around the text were missing. Completely.

When Don ran Kaypro's demo game that uses the 84's graphics, everything worked but the graphics (the game was much less interesting).

The Video Circuit

At this point let's take a quick look at the video circuit.

Figure 1 is the schematic diagram for the video circuit in the 84 Kaypros (the 1, 2, 4, and 10 with modem). The video circuit for the older 10 is virtually identical except that the IC numbers are different and a dozen or so small TTL chips take the place of U10 (a custom IC designed to replace those small TTL chips).

U16, the 6845 (or the 6545 which is identical) directs the whole operation. The Z80 processor can say "hello out there" by pulling down on the one VIDCS* line (the * means the device is selected when the voltage is low).

Then the processor puts a byte on the data bus that tells the video controller to, for instance, put the following byte in video memory. (That's a very common thing to say to a video controller.)

The 6845 then sets its ENable output (pin 23) high and its R/W output (pin 22) high (for a read operation). U10 then raises the enable line on U14 (pin 11).

U13 and U14 are latches. U14 grabs a byte from the Z80's data bus (when pin 11 goes high). Then when its output is enabled (pin 1 goes low) it writes that byte onto the video data bus. Thus, the Z80 can write data into video RAM.

U13 does the opposite. It grabs data from the video data bus and outputs it to the Z80's data bus.

You'll notice that there are two video RAM chips, U15 and U23. U15 holds the character byte (e.g. 41H for an 'A'). At the same location in U23 there's an attribute byte (reverse, blinking, underline, half intensity) for the 'A'.

So, after the Z80 taps on the 6845's shoulder (VIDCS* line) the 6845 tells U14 to grab a byte from the Z80's data bus. Then the 6845 puts a value on the video memory bus (determines the location on the screen) and tells U14 to output the byte to the video data bus. If the byte is a character to be displayed, then U23 (character RAM) is told to read the byte from the video data bus. If the byte is an attribute, then U15 (attribute RAM) reads the byte.

Busy Controller

Meanwhile, the video controller is running the scan address lines (pins 34-38) and it's selecting one character after another (with the video address lines).

It uses the video address bus to select the proper locations in the character RAM and attribute RAM for the location currently being written on the screen.

When you figure that in one scan line it's selecting and writing to the screen 80 different characters, and it's writing a scan line in 1/19,000 of a second, then you understand how busy the 6545, RAMs, ROM, video address bus, and video data bus are.

This part of the computer is constantly running full blast, even when the screen is just sitting there, blank.

Generating A Character

The controller is addressing the proper location in character and attribute RAM, using the characters and the scan data as an address for the video ROM (U9) and then turning the output from the video ROM into a carefully timed stream of ones and zeroes (lit or unlit pixels). The attribute bytes determine how this stream of pixel data is treated.

Reverse video is produced by reversing the bits in the stream (upside (continued on page 56)
Figure 1 - Kaypro 84 Video Section
KAYPRO COLUMN

(continued from page 54)

down, not backward). The circuit generates half-intensity by reducing the peaks of the video output.

Whew!
Things get particularly harried when the Z80 demands some attention, especially when the Z80 already has trouble meeting specs for getting information onto the data and address busses (a design feature of the 84 Kaypros).

So the Z80 is a bit slow, the 6845 a bit harried, and the whole thing has to work absolutely correctly or you get strange things in character and attribute RAM.

Back At MicroSphere
Our standard fixes for screen garbage have been to change the 6545 or 6845 to a 6545A or AE (a faster part). We've also had some success changing the 6116s to 6516s, and the video data bus. But it was soldered in place, so I looked around for another possibility. U14 was the only other suspect, and it was socketed.

Don noticed that U13 (the video-out latch) was a 74ALS373 rather than a 74LS373. The ALS part is much more expensive than the LS part so there must have been a reason that Kaypro used it (faster, newer, more powerful...). He found one in a drawer and we plugged it in to U14's socket. Voila! Perfect graphics, no extra characters. (I wish I had drawers like his.)

A quick comment about ICs: The letters that follow the "74" on ICs tell you how fast the chips are, how much power they draw, how they were made, and so on. The numbers that follow the letters (if there are letters) tell you what the ICs do and what their pin outs are. So, a 74373 (standard TTL) can be replaced by — a 74S373 (Schottky), a 74LS373 (low-power Schottky), a 74C373 (CMOS), a 74HC373 (high-output CMOS), and so on. Thus, we knew that a 74ALS373 would directly replace the 74LS373.

We even swapped back to the original 6116s, 6545, and Z80A, and the display remained as solid as a rock. No garbage. Full graphics. Everything. It's possible, therefore, that this might be the fix for those of you who've been frustrated in your attempts for clean, fast video. After all, 74ALS373s may be about $5 each, but for a clean screen...

Note: If you're getting garbage on the printer during screen dumps you might try replacing U13 with a 74ALS373 (like Kaypro did on our 10). If you have an older Kaypro 10, its U45 is the same as U14 and its U46 is the same as U13.

Kaypro Z-SYSTEM Now Available

Echelon proudly announces new era for 8-bit Kaypro users.
Begin using Z-System (ZCPR3 and Public ZRDS) as easily as booting your machine!

Set yourself free from that antiquated and inefficient operating system you now use. Z-System makes your Kaypro faster, friendlier, more intelligent and powerful. See columnist Ted Silveira's 2-part article, Dec. '85, Jan. '86 Profiles.

• ZCPR3 Environment allows features such as Named Directories, multiple command lines, ingenious batch processing, easy use of user areas to organize your files. Be capable of editing, proofing, and printing a document in a single keystroke!

• Public ZRDS Plus allows declaration of "public" directories, programs that use overlay files find them easily. Requirement to warm-boot after changing disk is eliminated, archive files, plus more!

• Completely compatible with your current CP/M™ programs.

• 70-page tutorial-style manual explains in detail how to make the most of Z. Numerous examples show use of Z elements: ALIASes, VILER, ZEX, flow control, MENU processing, Resident Commands, and the utilities.

• Over 60 utility programs included.

The Kaypro Z-System is available in five different versions, determined by model you use. When ordering, please identify your machine from this list:

• Pre-84 models
• 2, 4-84 or 2X models
• Kaypro 10 with D BIOS
• Kaypro 10 with F, G, or H BIOS
• Any model equipped with Advent TurboROM™

"Kaypro makes a nice computer—too bad it doesn't do graphics"

Make that didn't do graphics. SCS-Draw is here, bringing Macintosh-style graphics to the Kaypro computer. With SCS-Draw, you can use your Kaypro to draw a picture. Of anything.

The first true drawing program for the Kaypro, SCS-Draw turns your computer screen into a sketchpad on which you can draw detailed images like those shown here. These images can be saved on disk or printed on your dot-matrix or letter-quality printer.

There are many applications of SCS-Draw. For example, you could use SCS-Draw to design and print party invitations, banners, or technical diagrams. You could also use SCS-Draw to promote your business or design your company logo.

And the best thing about SCS-Draw is that it's fun to use — when was the last time you had some fun with your Kaypro?

SCS-Draw gives you a variety of useful drawing tools to choose from. You can draw a detailed image dot-by-dot, or do a rough sketch with straight lines. Need a compass? SCS-Draw can draw circles of any size, wherever you need them. For subtle shading effects, use one of 25 pre-defined patterns, or create your own.

Other features include block moves, four built-in fonts, and powerful print options like enlargement, indentation, mirror image and rotation. And with SCS-Draw's windowing feature, you can work on images much larger than your Kaypro's screen.

How good is SCS-Draw? Every day, we get unsolicited letters and phone calls from SCS-Draw users around the country — here's what they have to say:

- "Very easy to use."
- G.H., WPAFB, Ohio
- "Keep it up ... the program is great fun."
- J.S., Roseburg, OR
- "A pleasure to use ... user-friendly, fun, well put together."
- D.A., Kalamazoo, MI
- "Everything is bug-free!"
- D.C., Los Angeles, CA
- "Worth much more than its cost."
- B.H., Birmingham, AL

NEW for PrintMaster Owners: Now you can use your favorite PrintMaster images in your SCS-Draw drawings. The Image Extractor converts PrintMaster images to SCS-Draw image libraries. Price: $24.95

SAVE when you buy SCS-Draw and the Image Extractor together: Only $79.95

SCS-Draw can be used with most popular dot-matrix and letter-quality printers, including those from Epson, Star, Okidata, C. Itoh, Panasonic, IBM, HP, Diablo and Kaypro.

SCS-Draw, KAYPRO, Macintosh, CP/M and PrintMaster are trademarks of Second City Software, Kaypro Corporation, Apple Computer, Digital Research and Unison World, respectively.
Eight Booleans
In A Byte:
Bitwise Operations In Turbo Pascal

A boolean is a number that has two states, either true or false. So it's easy to represent a boolean with one bit. Easy, that is, unless the language you're using dedicates a whole byte to each boolean. So this article can save you some real space.

Also, just about everything we do at Micro C, whether it be hardware diagnostics, Z80 or 8086 Assembly language, or just speaking in hex (keeps the mailroom on its toes), requires an understanding of binary, hexadecimal, and decimal number systems.

Bill's thorough introduction to these number systems is as least as important as his Pascal routines, and it makes this an excellent article for you high-level folks who'd like to do some binary mucking with the rest of us.

Turbo Pascal can manipulate single bits of a byte. This article describes one application of this facility — using one byte to do the service of eight booleans — and shows how to implement it.

Since a boolean in Turbo is one byte long, you can save seven bytes. You might ask why anyone would want to save seven bytes; that's not much. But if you're building a data base that may contain 7,000 records, you're talking about 49,000 bytes, and that's a mouthful.

Review Of Numbers
Here's a quick review of theory for the newcomers. A byte is composed of eight bits, numbered 0 through 7, and each can be on or off, represented by 1 (on) and 0 (off), like this: 0011 1100.

The bits are shown in groups of four for clarity; bit 7 is the one on the far left (ultra liberal left) and bit 0 is the one on the far right (radical right). Each pattern of bits represents a number.

We've arbitrarily broken the 8 bits (they represent 0 - 255 decimal) into two groups of four bits.

Each group of four bits is a number ranging from 0 to 15 decimal. We created the hexadecimal number system so that we could represent those four bits as a single integer. Hexadecimal (base 16) numbers range from 0 to F, where 0 through 9 are the same as ordinary decimal numbers. Then follows A (10 decimal) through F (15 decimal). 10 hex equals 16 decimal. FF hex equals 255 decimal equals 1111 1111 binary.

(Note: a 'b' suffix means binary or bits — 'd', or no suffix, stands for decimal — 'h', or hex, stands for hexadecimal. In the range 0 - 9 there is no difference between decimal and hex.)

Thus, 0000b is 0, 0011b is 1, 0100b is 2, 0100b is 4, 1000b is 8, 1111b is F hex. 1111b = 8 + 4 + 2 + 1 = 15 decimal = F hex.

Turbo Operators
Turbo Pascal provides six bit operators — NOT, SHL (shift left), SHR (shift right), AND, OR, and XOR (Exclusive Or). They're disguised in the manual because it says they apply to integers, which are two bytes long (16 bits). But they allow you to set, clear, and test each bit in the integer. ("Set" means to make the bit a 1, "clear" means make it 0, and "test" means to see whether it is 1 or 0.)

Now, first I was talking about bytes and here I am talking about integers. What gives?

Well, type "Byte" is a subrange of type "Integer," so all this integer talk applies to bytes as well. The Turbo manual gives examples of these operations on integers which are somewhat cryptic if you don't understand what's really going on, so here's a more detailed explanation. I'll talk about bytes from here on, because that's what I'm interested in, but remember this applies to integers as well.

NOT, SHL, & SHR
NOT flips all the bits in a byte.

NOT 0010 0010 --> 1101 1101

SHL shifts the bits to the left a specified number of places. Bits shifted out of the byte to the left are lost forever in the "bit bucket" and zeroes replace the bits on the right.

1000 0001 SHL 2 --> 0000 0100

SHR shifts the bits to the right. It is just like SHL except in the opposite direction.

0000 00100 SHR 2 --> 0000 0001

These three, NOT, SHL, and SHR, operate on one byte. The next three, AND, OR, and XOR, have to do with the relationship between the corresponding bits of two bytes.

AND, OR, XOR
AND is defined as follows:

1 AND 1 --> 1
1 AND 0 --> 0
0 AND 1 --> 0
0 AND 0 --> 0

In other words, the result is 1 if and only if both corresponding bits are 1. In other words:

0101 0101 0101 0101 (5555h)
AND 0000 0000 1111 1111 (00FFh)
EQUALS 0000 0000 0101 0101 (5555h)

Note how the 0 bits in the second integer (actually, either integer) make sure that the corresponding bits in the answer are 0. The 1 bits in the second integer (again, either integer) mean that the bits in the answer are the same as the bits in the first integer.

OR is defined as follows:

1 OR 1 --> 1
1 OR 0 --> 1
0 OR 1 --> 1
0 OR 0 --> 0
In other words, the result is 1 if either bit (or both) is 1. This is used to set a bit to 1. If you OR a bit with 0 the result is the same as what the bit is, but if you OR a bit with 1, the result is always 1.

\[
\begin{align*}
0101 & 0101 0101 0101 (5555h) \\
\text{OR} & 0000 0000 1111 1111 (00FFh) \\
\text{EQUALS} & 0101 0101 1111 1111 (55FFh)
\end{align*}
\]

XOR is defined as follows:

\[
\begin{align*}
1 \text{ XOR } 1 & \rightarrow 0 \\
1 \text{ XOR } 0 & \rightarrow 1 \\
0 \text{ XOR } 1 & \rightarrow 1 \\
0 \text{ XOR } 0 & \rightarrow 0
\end{align*}
\]

In other words, the result is 1 if and only if the first bit or the second bit but not both is 1. You can use this to change a bit to its opposite or leave it the same as it was, without knowing what the original value of the bit is. If you XOR a bit with 1, the result is its opposite. If you XOR a bit with 0, the result is the same as the original.

\[
\begin{align*}
0101 & 0101 0101 0101 (5555h) \\
\text{XOR} & 0000 0000 1111 1111 (00FFh) \\
\text{EQUALS} & 0101 0101 1110 1110 (55AAh)
\end{align*}
\]

**Practical Bits**

Now for some practical applications. Instead of worrying about bytes and integers, what we really want to do is operate on individual bits.

The example program in Figure 1, SETBITS.PAS, contains two procedures, SETBIT and CLRBIT, and a function, TXTBIT. (If you know Pascal MT+ you’ll recognize that these are borrowed from that compiler, where they’re built-in procedures and functions.) In each case we use SHL to put a known bit in the position we want, then AND or OR the byte it’s in with the target byte to get the result we want.

SETBIT takes a byte and sets a specified bit to 1. You pass it the byte

(continued next page)
and an integer (n) between 0 and 7 (which bit to set). First the procedure creates an integer with the correct bit set by starting with the integer 1 (0000 0000 0000 0001b) and shifting it left n places. Then it ORs the result with the byte you passed to SETBIT. Here's a picture of SETBIT (FLAGS,2) where Flags is a byte.

```
0000 0001
SHL 2
-------
0000 0100
```

```
xxxx xxxx /*x* = unknown value)
-------
xxxx x1xx (now we know bit 2 is set)
```

CLRBIT sets a specified bit to zero. First the procedure shifts 1 left n bits (to the bit we want to clear), then reverses (with NOT) all the bits. Finally it ANDs it with the byte. Here's a picture of CLRBIT (FLAGS,2):

```
0000 0001
SHL 2
-------
0000 0100
```

```
NOT 0000 0100 (flip all the bits)
-------
1111 1011
```

```
xxxx xxxx
-------
xxxx x0xx (now we know bit 2 is 0)
```

TSTBIT tests an individual bit. First it shifts 1 left n bits, to the bit we want to test. Then it ANDs it with the target byte. We know the result for all the bits (except the one we're testing) will be zero, because ANDing a bit with zero yields a result of zero, so all that remains is to see if the value of the byte after the AND is zero or not, and we'll know if the bit is a 1 or a 0. TSTBIT (FLAGS,2) looks like this:

```
0000 0001
SHL 2
-------
0000 0100
```

```
xxxx xxxx
-------
AND 0000 0100
```

```
0000 0x00 (now we know all but bit 2 are zero)
```

---

**program setbits**;
{ To demonstrate operations on individual bits of a byte.
 Wm Meacham -- Revised: 12/28/85 }

```pascal
var
  flags : byte;  { the byte used as eight booleans }
  bit : integer; { which bit to set or clear }
  truth_value : char; { 'T' or 'F' -- which way to set the bit }

{ -------------------------------------------------------------- }

procedure setbit (var dbyt : byte ; n : integer) ;
{ sets bit n of a byte to 1 }
begin
  dbyt := ((1 shl n) or dbyt)
end ;

{ -------------------------------------------------------------- }

procedure clrbit (var dbyt : byte ; n : integer ) ;
{ sets bit n of a byte to 0 }
begin
  dbyt := ((not (1 shr n)) and dbyt)
end ;

{ -------------------------------------------------------------- }

function tstbit (dbyt : byte ; n : integer) : boolean
{ test bit n of a byte -- True if 1, False if 0 }
begin
  tstbit := not(((1 shl n) and dbyt) = 0)
end ;

{ -------------------------------------------------------------- }

procedure display_flags ;
{ display the boolean values of the bits }

var
  i : integer ;

begin
  for i := 0 to 7 do
    begin
      write (i) ;
      if tstbit(flags,i) then
        write (' TRUE ')
      else
        write (' FALSE ')
    end ;
  writeln;
end ;
```
begin { program setbits }

writeln ('This demonstrates setting individual bits of a byte.');
write ('Enter the number of the bit to set, '); writeln ('then how to set it -- True or False.'); writeln ('The bits are 0 through 7. Enter a number '); writeln ('outside this range to quit.');
flags := 0;
display_flags;
repeat
write ('Bit to set? (0 - 7) => '); readln (bit);
if bit in [0 .. 7] then begin
  repeat
    write ('Truth value? (T or F) => '); readln (truth_value);
    truth_value := upcase(truth_value);
  until truth_value in ['T', 'F']
  if truth_value = 'T' then setbit(flags,bit)
  else clrbit(flags,bit)
display_flags
end
until not(bit in [0 .. 7])
end.

{ ------------------------------------------------------------- }

Figure 2 - Code Fragment

var
  mask : byte;
  category : array [0 .. 7] of boolean;
  i : integer;
begin

  user enters Yes or No to select categories to print

  mask := 0;
  for i := 0 to 7 do
    if category[i] then
      setbit(mask,i);

  repeat
    { select records }
    { read a record from data file }
    if (mask and data.flags) > 0 then
      print_the_record;
  until eof(datafile);
end;

If the byte is not equal to zero, then the bit is a one. If it is equal to zero, the bit is a zero.

That's it! Now we have three useful additions to Turbo Pascal. The example program in Figure 1, SETBITS.PAS, gets user input and sets and clears bits accordingly. Then it writes out the values of each bit, interpreting a 1 as true and a 0 as false. (This is how Turbo implements a boolean, by the way. True is 0000 0001 and False is 0000 0000.)

Figure 2 is a fragment of code from a larger program that shows how you could use this facility. Each record in the data base has a byte that represents eight booleans, each of which is a selection criterion. They might be such things as "Environmentalist," or "Anti-Nuclear Activist," or "Democratic Party Member," etc.

If you want to print only those records that are Environmentalist, you enter Yes for Environmentalist and No for the other seven categories. If you want to print Environmentalists and Democrats, then enter Yes for both of these and No for the others. (Note that this would print all those who are Environmentalists or Democrats or both.) By constructing a selection mask and ADDing it with the flag byte in each record, you can easily make the selection.

Final Words

These procedures work equally well with integers; just declare the variable dbyt as an integer. Using integers gives you 16 booleans instead of 8.

Credit where credit is due: I figured out how to implement eight booleans in a byte by using powers of 2 to set the bits. My friend Steve Elliot showed me Setbit, Clrbit, and Tstbit. Thanks, Steve!
A lot has been written about hard drives. Most of the articles I've seen have been the "gee whiz" variety. Gee whiz, drives are getting cheap. Gee whiz, drives are holding a lot of data. Gee whiz, a lot of people are buying drives. But those articles usually don't deal with what happens when a drive dies. What is the seller's responsibility? Where do you turn if the seller can't (or won't) help? Which manufacturers are supporting end users and which aren't.

A hard drive is probably the most expensive accessory you'll purchase for your system. It's also probably the most difficult to maintain. After all, what other mechanical repairs, and works with a controller board (which has its own problems). Stir in a dearth of servicing. A device operates at such tight tolerances, requires a clean room for mechanical repairs, and works with a controller board (which has its own problems). Stir in a dearth of servicing information for either the controller or drive and you can get a real queasy feeling.

You find out quickly that you're really dependent on support from the dealer you bought the drive from. In some cases he's the only support you have. In other cases, however, the manufacturer is willing to help. I'll be covering both those situations.

In researching this article I spent lots of time on the phone finding out: first, just what are winchesters (or hard) drives? Second, who is supporting them? What is the seller's responsibility? Where do you turn if the seller can't (or won't) help? Which manufacturers are supporting end users and which aren't.

How Winchester Work

Hard disks come in all sizes and flavors: the early standard was 5 megabytes. Now the standard is 20 and moving rapidly toward 30 and 40 Meg. If you've been checking prices, you will have noticed that you can pick up a 20 Meg hard drive for little more than the price of a 10.

In fact, many manufacturers aren't even making 10 Megs anymore. Here's the reason: if you could make a 20 for almost the same price as the 10, would you make the 10? Of course not. Besides, anyone who can use 10 megabytes will sooner or later (usually sooner) find a use for 10 more.

Stepping Out

A 10 megabyte drive usually contains two aluminum platters. Each platter is covered, both sides, with an oxide coating. Four heads (one per surface) are connected to a single actuator arm and move in tandem across the platters.

In most drives, the actuator arm is driven by a small stepper motor (the system is similar to the stepper/head arrangement on floppy drives). For these drives, the step rate (time to step from one track to the next) is between 1 to 6 ms. (Most floppy steppers are happy at 3 or 6 ms.)

The step time is very important because most winchesters have over 400 tracks (compared with 40 tracks for 48 tpi floppies). And, as disk and head technologies improve, manufacturers are crowding more and more tracks (more and more data) onto the same number of platters. Meanwhile, stepping out (especially all the way out) is taking longer and longer.

Positioner technology is also improving. Many of the larger capacity drives (30 Meg and up) have fancier steppers. Some of them are mechanical, but generally the newer, faster units use voice coils to move the heads and optical sensors to keep track of position.

Voice coils are simpler (less to wear out), more accurate (because of optical positioning), and faster (step rates in the micro second range). They are also more expensive (of course).

Hard Disks

Most drives contain either 3 1/2" or 5 1/4" platters. We'd been questioned by readers about manufacturers "sneaking" 3 1/2" platters into a drive designed to hold 5 1/4" platters. We wondered if manufacturers weren't pulling the wool over buyers' eyes. But according to Ron Sheltzus, director of marketing for Microscience, a major hard drive manufacturer, the capacities of the two sizes of drive are virtually the same. In other words, if you buy a 20 megabyte hard drive, it doesn't matter if it contains 3 1/2" or 5 1/4" platters — you're still going to have a 20 megabyte hard drive.

Ron also explained that since the 3 1/2" are newer, some are more technologically advanced. They also weigh less, and have fewer components to go wrong. But then, the track and bit densities are higher. Manufacturers have gotten into trouble trying to squeeze higher and higher data densities out of the old (media, head design, and head positioning) technologies.

Interleave Factor

One more tidbit about access speed. Nearly all drives rotate their platters at 3600 RPM, so everyone's equal there. However, let's say the controller wants to read sectors 1, 2, 3, and 4. And let's assume the controller needs some time to deposit sector 1 in memory before it's ready to read sector 2. If the sectors are all lined up 1, 2, 3, 4, 5, 6... on the track, then the beginning of sector 2 would have already passed under the head by the time the system had finished with 1. Thus, the controller would have to wait a complete revolution of the disk between each sector.

Well, lining up the sectors 1, 2, 3... is called an interleave of 1 (it's really no interleave at all). An interleave of 3 means that the sector labelled sector 2 would show up 3 sectors after sector 1, sector 3 would show up 3 sectors after 2 and so on. It takes three revolutions of the disk to sequentially...
read all the sectors when a track has an interleave of 3. A common interleave for hard drives on MS-DOS systems is 7.

Interleave is generally a function of computer speed, not drive or controller speed. If the computer is fast enough to read and write sequential sectors with no interleave then, of course, it has a big advantage.

When you're formatting a hard disk you can usually select the interleave. If you start with an interleave of 7, you may find that disk access gets faster and faster as you reduce the interleave until you reach the optimum value. Then, reducing the interleave just one more notch drastically increases access time.

Taking The Plunge
Now that we know a little about hard disk drives, it's time to start comparing manufacturers, their attitudes, and policies.

First of all, you seldom buy a drive directly from the manufacturer. Most manufacturers prefer selling only to dealers or OEMs (Original Equipment Manufacturers. IBM is an OEM, which means it buys the components for its systems from component manufacturers, puts them together, and sells the final product under the IBM name.)

Chances are you'll be buying your hard disk from a dealer, and if you've perused the pages of Byte, Computer Shopper, Micro C, or practically any other publication, you'll see the dealers' ads hawking their wares. Literally dozens of them sell hard disks ranging in price from about $300 for a house brand 10 Meg on up to $2500 or so for an Iomega Bernoulli box.

As you go through the list of 800 numbers and talk to salespeople, you'll quickly learn which dealers know what they are selling and which don't.

You'll want to ask if the advertised price includes cables and controller card. Drives advertised for PCs and XT's nearly always include cables and controllers. Drives listed for the ATs don't, because most AT-style systems have controllers built in.

(continued next page)
HARD DRIVES, THE SERVICE PERSPECTIVE

(continued from page 63)

Warranties
You'll also want to ask about warranties. Almost all manufacturers offer a one year warranty on the drives. Some, like Maynard, offer a one year warranty on the drive and a longer warranty (up to five years) on other parts, such as the drive's electronics and/or the controller card. However, dealers can substantially limit customer support.

For instance, let's say Rodime offers a one year warranty to the original purchaser. Rodime sells only to dealers and OEMs. A dealer buys a Rodime drive, then sells it to an end user and offers only a 6 month warranty. Let's say there's a problem with the heads 8 months after the end user buys it.

Since the drive is no longer under dealer warranty, the user will have to pay for repair or replacement. The dealer, in turn, sends the drive back to the manufacturer and gets a new one free. This scenario assumes, of course, that the dealer sold the drive immediately after getting it from the manufacturer. It's very possible, however, the drive sat on a shelf in the dealer's warehouse for who knows how long before being sold. So when a dealer puts a 6 month warranty on the drive that originally had a 1 year warranty, he may actually be on the hook during part of the 6 months.

A word more on warranties: if you return a drive that's obviously been abused, or there are components missing, or you'vemonkeyed with the seal, kiss your warranty goodbye. No manufacturer will honor the warranty in such a case. However, rewiring something on the drive, using different cabling, or something of this nature won't necessarily invalidate a warranty.

For instance, Priam, a hard drive manufacturer, will fix anything that's original to their drive. So if you alter the wiring, and that's the cause of the problem, you're out of luck. But if it's original components causing the problem, Priam will repair it.

Also, some manufacturers have been installing G-force detectors inside their units. If the drive is dropped, the detector will be permanently altered, and the warranty will be voided.

Drive Problems
The key to winchester technology is that manufacturers can pack lots and lots of whole bits (no half bits, because of the sharp edges) into a very small space. To do this manufacturers place a narrow head very close to (but not touching) the media. In fact, the head floats on a cushion of air about 1/10th of a hair's width above the spinning disk. Any dust (such as a particle of tobacco smoke) on the platter would be ground between the head and the media, probably damaging the media.

In fact, even the head could damage the media if the two touched while the platters were spinning.

So, the platters and heads are sealed in a dust free case and all repairs inside this case have to be done inside a clean room.

One of the reasons drive manufacturers don't like to deal with the public is that dealers should be better able to weed out the problems caused by defective controllers (and even...
then, about half the "defective" drives returned to the factory work fine).

Attitude Toward The User
Manuifacturers' attitudes toward dealing with the end user vary greatly. From what we can tell, all manufacturers service their own drives. But most of them, such as Micropolis, prefer not to deal with the public, and will do so only as a last resort. Micropolis wants users to return drives to the authorized dealers, which are Hallmark on the East Coast and Wiley on the West Coast. But if for some reason the dealer is totally unhelpful, Micropolis will talk to you.

Other manufacturers, such as Microscience, have a strong relationship with their dealers and OEMs, and insist that users return drives to those dealers. However, Microscience makes up for this seeming inconvenience to the user by allowing authorized dealers (Wiley and Pioneer) to give the end user the full 1 year warranty Microscience offers to the dealer. Virtually none of the other manufacturers we talked to in preparing this article offered that kind of warranty to the end user. But remember, to get that warranty, you must buy through either Wiley or Pioneer.

Another plus for Microscience: they've recently negotiated with FRS, Inc., a large repair service in Sacramento, CA, to be an authorized Microscience service center. Repair rates vary depending on the problem. For tech questions, the end user has the options of asking the dealer, calling FRS (916) 920-1107, or calling Microscience directly (number listed below).

Maynard takes a slightly different approach. Though it sells through dealers only, you have the choice of taking a problem drive back to the dealer or shipping it directly to Maynard. It claims to provide 48-hour turnaround on non-clean room repairs (things like stepper motor, controller board, or disk drive board). Maynard sends clean room operations back to the manufacturer (Maynard is a rebranding company), but meanwhile they will immediately ship you a new drive.

Xebec is another rebranding, and like Maynard, Xebec provides very decent support for the end user including an 800 number directly to the technical department (see list below).

Note: Maynard does not manufacture its own hard drives. Instead, it tests and uses such drives as Rodime, Seagate, Fugi, etc. in its systems. When sold, these drives may or may not bear the original label. Maynard claims that because it is constantly testing new hard drives, and that the quality of drive a manufacturer produces varies from model to model, it doesn't always stick to the same manufacturers. So if two people buy a Maynard drive, one may get a Seagate, the other a Rodine. But both will receive a controller board manufactured by Maynard expressly for that drive.

Thus, for simple technical questions, manufacturers generally don't mind a call from the end user, and seem pretty willing to help the little guy. However, virtually all of them, except Maynard, want you to take a problem drive to the dealer first, and only if that proves unsatisfactory, then to contact them.

Following is a list of some major manufacturers, along with their addresses and phone numbers.

Control Data *
8100 34th Ave. South
P.O. Box O
Minneapolis MN 55420
(612) 853-8100
Repair Center, Oklahoma City
(405) 324-3160 (tech help line)

Maxtor *
150 Riveroaks Parkway
San Jose CA 95134
(408) 942-1700
(408) 435-3354 (tech help line)

Microscience *
21123 Nordhoff St.
Chatsworth CA 91311
(818) 709-3300

Miniscribe *
1871 Left Hand Circle
Longmont CO 80501
(303) 651-6000

Quantum *
1804 McCarthy Bl.
Milpitas CA 95035
(408) 262-1100

Tandon
20320 Prairie St.
Chatsworth CA 91311
(818) 993-6644
- no tech help number is available to the end user
- authorized dealers offer the end user a 90-day factory warranty on most winchesters

Xebec *
2055 Gateway Pl.
San Jose CA 95110
(408) 287-2700
(800) 982-3232 (tech help line)
* offer 1 year warranty on drives to original buyer

If you have a tech help question about a drive manufactured by one of the above companies, call the tech help number (if one is given) or call the company's main number and ask for technical assistance. You'll be transferred to the right department.
Purchasing And Installing A Hard Drive

What Your Mother Didn’t Tell You About Winnies

One thing Becky and I found out as we started poking around the hard drive arena is that there are a lot of drives out there — and a lot of people willing to give us information about those drives.

But most of the information wasn’t very interesting. The really meaty stuff, like whose drives are dropping like flies, and whose controllers don’t work with whose drives, wasn’t volunteered. But, by listening between the lines, we managed to come up with some pretty solid suspicions. (Stay tuned so we can all be suspicious together.)

Finally, a number of you have wondered what it takes to attach a winnie to your clone (about $500). The last part of this article covers what to do on the morning after you’ve laid out all that cash.

I’ve read a lot of articles, talked to a lot of people. I’ve asked a lot of questions, all of them boiling down to: which winchesters are worth buying?

Here are the questions I asked: Which are the most dependable? Which are the easiest to get serviced if necessary? Which are the best value?

The people who knew most of the answers about reliability and serviceability were the repair shops. They weren’t talking though, because they were afraid of being cut off from their sources of parts and schematics.

However, during my reading and interviewing I did learn the following:

1. If you buy a CMI drive, lotsa luck. IBM is no longer purchasing CMI drives (though they never admitted that the drives had problems). Anyway, CMI may disappear any day, but their drives (especially the pull-outs) will be hanging around the surplus market for quite a while.

2. The only thing worse than a CMI drive is an early Tandon. We had several of the early Tandons, and they nearly shut down the Micro C editorial department, which makes them even worse than CMI drives.

3. Hard drive failures aren’t necessarily caused by problems with the drives themselves. The problems often result from controller or software incompatibilities, or from controller board failures. Xebec technical folks indicated that about half of the drives they receive from field engineers work just fine.

4. When a hard drive is defective, 60 percent of the time the problem is inside the bubble (the sealed area where the platters and heads reside). Of these bubble problems, 75 percent are head related, 25 percent are media (platter) problems.

A Gold Mine

Allyn Franklin is my drive contact. He gets first shot at any questions I have (lucky guy). He suggested I talk to Stuart Eaton, a winchester drive repairer.

I did, but Stuart couldn’t tell me much. Oh, he knew it all right. But there were those non-disclosure agreements and so on.

But he did turn me on to David Claridge, a senior technology analyst for Hambrecht and Quist of San Francisco.

All Business

David is more a business type than a technical type, but I must say that if I were planning to purchase stock in a hard drive company I’d call him first.

He didn’t want to say anything bad about anyone, but it soon became apparent that he had reasons for not discussing some folks.

I asked him to name the manufacturers (not the re-branders) he was comfortable with. He listed the following:

Control Data
Seagate
Maxtor
Micropolis
Miniscribe
Priam
Quantum
Rodime

He noted that the first four are U.S. companies, but they are having their drives built somewhere in Asia. Micropolis and Priam, on the other hand, are building drives here. Quantum is partly here, partly in Asia. Rodime is manufacturing in Florida and in Scotland (an interesting country to visit, but can they build drives?).

Added to that list are four Japanese firms he thinks are, or will soon be, major players.

Hitachi
Fujitsu
NEC
Siemens

Increasing Reliability

I told him I was concerned about drive reliability. He appeared unconcerned about that.
“Reliability has been improving. A couple of years ago you were lucky to get 10,000 hours between failures (MTBF). That figure has risen to the point that failures aren't much of an issue.

“As long as the use isn't very heavy (8 hours per day intermittent use) there isn't much difference between the low end drives.

“Look at Seagate: they've shipped an incredible number and they've been solid. (I think the number is about 1.25 million drives for Seagate. MTBF for the ubiquitous 20 Meg ST225 is reportedly 20,000 hours.)

“Then there are the mid-range drives. They have faster access times but are still stepper actuated, like Quantum.

“On the high end is the 85 Meg Micropolis drive. It's probably the most reliable drive on the market. It has an MTBF of 50,000 hours. (That's 5 1/2 years at 24 hours per day.) A truly outstanding drive. They're built in the U.S., but the company is ramping up in Singapore.”

A Commodity Product

David noted that during the '60s and '70s price/performance was improving about 25 percent per year. During the '80s that's risen to between 40 and 60 percent per year. The reason is that winchesters have become a volume business. Thus:

1. There's capital to research new technology.
2. There are intense competitive pressures from offshore manufacturers.

The 3 1/2” Winchesters

David predicts that the 3 1/2” winnies will take over the low end of the market at some point. In fact, the smaller drives are only one of the ways manufacturers are cutting costs. The latest 5 1/4” Seagate 10 Meg reportedly has only two heads and a single platter, down from four heads and two disks just a few months ago.

The move to smaller, simpler drives is made possible by changes in head and media design. For instance, manufacturers are moving to plated media.

(continued next page)
onto an aluminum platter (just like chrome used to be plated onto steel bumpers).

Oxide media, the current favorite, is made of tiny particles of iron oxide suspended in a polymer binder rather than solid magnetic material. Data density is much lower with the oxide technology.

Coming Shortly

David was particularly excited about the upcoming announcement of the new 40 Meg Seagate drives. They’re supposed to cost no more than the current 20s (that would put them at about $350 each without controller) and have a 40 ms average access time (compared with 80 ms for the 20s).

What’s surprising everyone in the industry is that Seagate will be using a common little stepper motor. But it’s a faster stepper than anyone would have imagined only a short time ago.

So

What should you purchase? A drive and controller that work together, obviously. Something dependable, obviously.

We have five 20 Meg Seagates in and about the office. They’ve been solid, quick (not super-quick, just quick) performers. We’ve got them connected to both Western Digital and DTC winchester controllers. Don (at MicroSphere) thinks the DTC controllers are best. I can’t say. Ours all work fine.

Best of all, they’re cheap. Really cheap. Just about every $550 (and under) 20 Meg winchester and controller board package for the XT’s and clones contains a Seagate 225 (20 Meg) and either a DTC or Western Digital controller.

Firing Up A New Winnie

This is the part a lot of folks are worried about. Connecting it, formatting it (if necessary), and copying over MS-DOS so you have a runnable, bootable winnie.

It isn’t hard, believe me. Manufacturers have taken practically all the sweat out of the operation (you can even add a second winnie to the same controller).

Winnie A La Formatte

1. Plug the controller board into a free slot. (Use any slot except the one closest to the power supply — it’s reserved for adding ROMs to the system.)

2. Mount the winnie in the computer. (I leave the details as a problem for the student.) It’s best not to have anything on top of the winnie because it can get a bit heated if someone’s cutting off its circulation. Normally the drive will arrive jumpered correctly (jumpered as drive 0). If you’re at all queasy about what’s going on, you might stick it into place but not bolt it down until it’s working.

3. The 34-pin ribbon cable runs between the controller and the drive (cables usually come as part of the package). Note that the red edge of the cable connects to pin 1. Look for a small ‘1’ and/or ‘2’ on the controller board near one end of the 34-pin male connector. These mark pins 1 and 2. On the drive you’ll see a slot cut into the 34-pin edge connector. The slot is nearest the pin-1 end.

MULTI-USER 68000 SINGLE BOARD COMPUTER

FEATUREING:

• 8 MHZ 68000 CPU
• 2 Mega Bytes RAM
• Memory Mapping / Manager
• HD 64180 Co-Processor
• 8 Serial Ports
• Floppy Controller
• SCSI / SASI Interface
• Parallel Printer Port

Price: $1500.00

HAWTHORNE TECHNOLOGY

8836 S.E. Stark, Portland, Oregon 97216
(503) 254-2005
4. Connect the 20-pin ribbon cable between the controller and the drive. Again, the red edge is pin 1. Watch for the '1' or '2' on the controller and the slot in the drive's connector. There are two 20-pin male connectors on the controller, one for each of the two drives it will control. Use the male connector closest to the 34-pin connector for the first (or only) drive. This 20-pin interface carries data back and forth between the drive and the controller.

5. Fire up the system. You'll need to have a bootable disk in drive A. On that disk you'll need DEBUG, FDISK, and FORMAT from the MS-DOS master disk.

6. From the A> prompt enter DEBUG. After DEBUG signs on enter:

```
g=c800:5
```

This gets you into the hard disk controller's formatting utility (it's in the ROM). The utility will ask for the hard disk type (it's '2' for the ST225) and for the interleave ('6' is a good choice for 4.77MHz clones). Then the controller does the low level format.

7. Run FDISK. Don't be confused by this routine. Just select the defaults. (It's just making sure you want to use the whole drive under MS-DOS.) If the disk was properly formatted in step 6 then FDISK will be happy.

8. Now, enter:

```
FORMAT C:/s
```

When this finishes you'll have a winnie that you can boot, save files on, and otherwise enjoy.

More Information
When you are purchasing your drive and controller be sure to let the seller know you want the installation manuals. They get them. You need them. If they say they don’t have them, or won’t send them, deal with someone else.
Languages, The Usual, The Unusual

This time Steve covers the availability of unusual languages, Japanese software, property management, and REC.

One morning some months ago while I was waiting for the 6:26, a strange little man came up to me and asked, "Whatever happened to Unix?" What a way to start a day! All day long I pondered the question. Had this wave of the future passed me by while I wrestled with CP/M and DOS?

I didn't know then and I don't know now the answer to the question. My suspicion is that as more and more of the features of Unix appear in DOS, Unix for the micro will be unnecessary. However, maybe you too have noticed a trend, and perhaps that trend is where computing will be in the not too distant future.

Certain languages seem to be "in" — especially C. (The buzzword set is hot on anything even remotely connected with Artificial Intelligence. But as long as you're not writing in either BASIC or Assembly language, you're au courant as far as computing is concerned.) Thus, while Unix may not be "in," C (the heart of Unix) is definitely "in."

Language Software
The starting point for learning C is the public domain Small C. A Small C compiler for Z80 CP/M is available on SIG/M volume 224. Small C for an 8086 machine is on SIG/M 149. The latest version of Small C for MS-DOS is on PC/Blue 192.

Small C is just that, a subset, and once you leave the initial learning phase, chances are you'll want to go into a commercial package. However, if you want to polish your skills, it may well pay to stick with and try adapting some of the programs in the library to Small C. Whatever you decide, there's a lot of good C code in the SIG/M library.

XLISP in CP/M 80 on SIG/M 118 and CP/M 86 on 153, written in C, is both a language and an opportunity to study some good C code. Same thing with PISTOL on 114. You might also get some ideas from CITADEL, a BBS system, on SIG/M 150. YAM in CP/M 86 on SIG/M 158 and CP/M 80 on 183 are modern programs in C. Want to see how a pro writes a text formatting program in C? Try ROFF on 174. Doing an editor in C? Take a look at SIG/M volume 76.

Unless you enjoy reinventing wheels, the availability of source code with these public domain programs not only lets you customize programs, but also lets you see how others write in C. For example, take a look at SIG/M volume 271, "Some C Programming from Japan." The same holds true with the Australian C code available throughout the collection.

Other Languages
Some other languages available in the SIG/M library include TIMCMP & PIDGIN (43), SAM76 (53), Little Ada (92), JRT Pascal (129), TINCMAP and META in CP/M 86 (159), Concurrent Pascal-S (162), FORTH (204), META4 for 8080 (207), COBOL (209) and Small PROLOG (242).

The PC/Blue library has FIG-FORTH (54), XLISP (146), ESIE — Expert Systems Interference Engine (176), LISP (188), and ICON — a SNOBOL like language (192), in addition to Small C on 192.

New From Japan
I did a paper in graduate school that involved a queuing simulation of the school cafeteria to demonstrate why the professor should let the class out early for lunch. The Japan Users Group (CP/M) recently contributed a four volume General Activity Simulation Program (GASP) to SIG/M (volumes 256 to 268). For those of you who are also into queuing, this set is quite a find. Unfortunately, it's written in FORTRAN (both for CP/M 80 and CP/M 86). However, it works and it works well.

Also from Japan, there's Statistical Distribution and Multiple Regression (SIG/M 269). The code is FORTRAN 80, which again is unfortunate. SIG/M 270, however, has some Turbo Pascal programming from Japan which shows that Borland's influence is spreading.

Property Management
For at least a year there's been a stack of blank disks on my desk from people who wanted a copy of the SIG/M Property Manager program (SIG/M volumes 199 and 200 and PC/Blue 75) in dBASE3. I'd promised to write a dBASE version, but, well, you know how much work it takes to do it right.

I finally made the time to do it. It seems to work quite well and will be out in the PC/Blue library before the end of the year. If you manage real estate and have a PC it's a dandy program that produces everything from rent bills to financial statements. If you can't wait until its official release, send me two 5'' disks and return postage and I'll send you a copy. One disk has the program for dBASE III+, the other is compiled with Clipper.

REC Surfaces Again
When we were talking about languages, perhaps we should have mentioned REC. While not a language by strict definition, Regular Expression Compiler from Prof. Harold Mcintosh at the Universidad Autonoma de Puebla, Mexico, is the closest thing to a language.

REC appears in quite a number of places in the SIG/M library. The most current versions, with and without floating point math, in CP/M 80 and CP/M 86, appear on SIG/M 214 and...
215. However, using REC and its adjunct CNVRT (see SIG/M 215), Mc-
Intosh has done quite a number of
other programs including Maze dem-
tractions (245), a giant game of Life
that plays out over a period of days
(246), multi-column and multi-file
printing (254), a CP/M 80 to CP/M 86
translator (203), etc., etc., etc.

Anyway, the school finally bought
the good doctor a PC (a Columbia, I
think) and he now has a version of
REC for MS-DOS. It’s available on PC/
Blue 211 and 212. This is not for the
appliance user. However, in the hands
of a hacker REC can be one of the
most powerful tools one can get for
the PC.

New Demon Shareware
A number of programs of interest
are in the new batch of PC/Blue
releases. There is a Ham Radio pack-
age on 219, and an RBASE clone called
PC-RIM on 204, 205, and 206. (RBASE
is a pretty good data management
system. PC-RIM is similar, although
slower.)

PC/Blue 203 has a pretty good pro-
gram in Turbo Pascal called Reliance
Mailing List. To show you how good I
think it is, Bill Meacham (the author)
sent SIG/M two copies of the pro-
gram. One was for the PC and the
other for CP/M. We gave Hank Kee
his copy for PC/Blue and will put out
the CP/M version very shortly. I’m
still pondering over the fact that the
program requests a modest donation.
Do we leave in the request, or recom-
pile the code and delete it?

Version 2.6 of PC-Write complete
with laser printer support is on PC/
Blue 202. MR. BILL, a billing program
by Dave Alexander, is on 207 and 208.
Dave’s CK, a time tracking program, is
on 209 and 210. Hank Kee and I
disagree on these programs. Dave sent
them to me for Hank. While they do
perform their function, I thought the
screen displays were too poor to be
used in a professional environment. I
also thought that source code is almost
mandatory for this type of program
because of the need to customize.

Maybe this gets to the heart of the
shareware concept. (As an aside, one
letter from a reader asked me to define
shareware. Shareware is the software
equivalent of a bag lady. She sits there
and asks for a handout and then
curses you if you say no or ignore her.)

I’m sure that Dave Alexander wrote
this program for his own use in his
law practice. Since I also practice law
and write software, I can assure you
that law is much more lucrative. In
the world of CP/M, people write software
for their own use and then let their
fellow computerists have a copy of it
— by donating it to the public domain.
The IBM people think every program
they write is another Lotus. Come on,
guys and gals — let’s release the
source code in the CP/M tradition and
forget about trying to be a bag lady.

Of Toxic Gas, Pianos, Etc.

When I was a kid, my piano teacher
quit on me. Perhaps that is why
PianoMan on PC/Blue 216 played flat
on my computer. I guess a tone deaf
person has a non-musical computer. It
does look cute, but I suspect that the
quality of your audio is a factor in
using it. PC-Key-Draw version 3.0 pro-
ducing keyboard to screen graphics is
on 217. There’s an update to Genealo-
y on Display on 218 and a Toxic Gas
Emergency Program from Canada is
on 220.

Volume 201 contains something
called Still River Shell, a file and
directory management program, and
Calc2, an extended precision calcula-
tor. You can do without both very
easily. However, the Wagner File Utili-
ty is also on the disk. It gives you an
overall view of your hard disk. Vol-
umes 211, 212, and 213 contain some-
ting called AnalytiCalc. If you don’t
have a spreadsheet, you could use this.

Other Odds And Ends

I had noted earlier that the Everex
Graphics Edge card in my PC did not
support screen blanking. Mark Hersey
of Hersey Micro Consulting, Inc. sent
over a copy of Fansi-Console to solve
the problem. If you have the same
problem, the program is available on
PC/Blue 161.

I watched Bill Cull demonstrate a
ZCPR2 loader at the Trenton Comput-
Pro. However, if you have a Kaypro
and haven’t the skills to install ZCPR,
you may want to contact Bill at 104
Burleigh Drive, Ithaca, NY 14850. He
also promised a copy of a universal
hard disk driver for the SIG/M library.

Getting The Software

Magazines are filled with advertise-
ments for public domain software for
purchase or lease. Most of those ad-
vertising are commercial companies.
However, as long as their price is
competitive and their service satisfac-
tory, they are a good source of soft-
ware. And they usually have a variety
of formats.

You can get SIG/M and PC/Blue
volumes through local distributors or
BBS systems. They are not-for-profit
computer clubs and provide these vol-
umes as a public service. Or you can
order directly from SIG/M (the Ama-
te Computer Group of New Jersey)
or PC/Blue (the New York Amateur
Computer Club).

SIG/M volumes are available on 8"
SSSD disks for $6 each ($9 foreign)
directly from SIG/M, Box 97, Iselin, NJ
08830. They are also available in most
5" formats. The charge for 5" disks is
$7 per volume. However, for SSSD
formats, or any format which requires
more than one disk, please add anoth-
er $2 per volume. Printed catalogs are
$3 each ($4 foreign).

PC/Blue volumes are $7 each ($10
foreign). The printed catalog is $5.
Both are available from the New York
Amateur Computer Club, Box 100,
Church Street Station, New York, NY
10008. Both groups have a disk catalog
(Volume 0) available at the price of a
standard disk volume. This catalog
volume is usually more current and
easier to get than the catalog.
A Z80 Slave Board

The TurboDos reviews continue. Plus, Dave covers sources of S-100 information, S-100 boards, and discusses power supply filter problems.

In this column, I'll be answering some questions from the reader mailbag before it explodes all over my desk. But first, let's take a quick look at Earth Computers' S-100 Z80 TurboSlave I board.

The TurboSlave I

Likely by its users, the TurboSlave I can run not only as a TurboDos slave, but also as just about anything else that's written in 8080 or Z80 code.

A built-in on-board monitor allows the board to be immediately powered up in an S-100 frame to take commands from a terminal connected to one of its serial ports. The board can boot immediately in a TurboDos environment and can be told to boot other operating systems from the host machine with a simple monitor command.

Also built into the TurboSlave I is a set of diagnostic routines that will automatically check the board and flash an error message (or an 'OK' message) on an on-board LED. The tests include checks for a failed EPROM, bad RAM in either bank, the FIFO buffer, and the DUART serial I/O.

Here are the specifications:

* 8 MHz Z80/H processor
* 64K or 128K of on-board 4164 dynamic RAM with up to 16K common area and parity error detection
* 2732 EPROM monitor (also accepts 2764 or 27128 EPROM)
* Two serial ports with software programmable baud rates (2681 DUART)
* A 2Kx8 FIFO buffer between the host processor and the TurboSlave I's port-mapped I/O
* On-board vectored interrupt capability to on-board Z80 CPU
* Built-in diagnostics (as mentioned above)
* Uses only two 8-bit I/O ports in the host processor
* Fully IEEE-696 compatible (of course)
* Requires 1.25A at 5V, 100mA at -16V maximum

Like most TurboDos slave processors, the TurboSlave I requires only two serial ports to talk to the host processor. Information is provided in the manual to allow the slave board to load just about any program or operating system from the host processor via the port-mapped interface as if it were a simple I/O device (which it actually is).

The TurboSlave I comes with a complete manual, including full schematics, a detailed theory of operation, sample drivers, and a disk full of TurboDos drivers and additional documentation.

Reader Mail

Several times, in past S-100 Bus columns, I've recommended to those interested in the S-100 IEEE-696 bus a book called "Interfacing to the S-100 IEEE-696 Bus" by Sol Libes and Mark Garetz. This book is really the definitive source of S-100 bus information and even includes a copy of the IEEE-696 standard (as it was proposed when the book went to press). Many readers have written to tell me that they are unable to get the book from their local book stores. Unfortunately, I am told, it is out of print (and not likely to be reprinted), and therefore unavailable to new S-100 users who would almost certainly be interested in its contents.

Fortunately, copies of the IEEE-696 standard are still available from the IEEE, so at least the technical facts about the S-100 bus are still in print (and of course, you can always ask about it in "The S-100 Bus").

S-100 Board Sources

Many readers have also asked for sources of S-100 components, including boards, frames, and replacement parts. Well, there are many sources, although some of the old "standard" suppliers may have dried up. Readers interested in information about S-100 items can check those periodicals that support it (Micro C comes to mind as one...), write to or call S-100 manufacturers (there are still lots of them), or check with some of the many mail order houses that sell S-100 (for example, Priority One, Jameco, etc.).

TurboDOS + SBCs

A very interesting letter from John G. Hall brings up the question of whether it's possible to use a single board computer (SBC) in place of a slave processor in a TurboDos system. In his letter, John also mentions that he thinks SBCs linked to a TurboDos system could be of great use in machine control applications where each might control a sophisticated production machine.

Well, I've never seen a stand-alone SBC linked directly into a TurboDos-based machine, but it's certainly possible. Since most slave processors used in TurboDos-based systems use only a few simple I/O ports, connection of an external SBC could be done with ease, as long as the proper transmission lines were established between the TD system's bus and the SBC.

There are already networking systems available for TurboDos that use twisted-pair or coaxial links between machines (even IBM PCs can run on them), so it would be possible, I suspect, to interface almost any SBC the same way. Peak Electronics (I believe) is even working on a 68000 SBC for TurboDos.

The Ripple Effect

Something to watch out for in older S-100 frames is the failure of the large
filter capacitors that are used in the unregulated S-100 power supplies. Unlike the rather easy-to-detect failures in most other computers that use regulated power supplies, problems in the supply of an S-100 machine can cause insidious troubles that will lead to hair pulling.

Common symptoms of S-100 power supply failures include erratic disk operation, unpredictable system crashes, programs that begin behaving strangely, and very often, serial communications failures.

Interestingly, the capacitor that most often fails and causes these problems is the one that filters the -16 volt supply. Perhaps this is due to the fact that this supply is usually the least used, but often has a very large filter capacitor.

Very often, monitoring the suspect supply with a DC voltmeter will NOT find this kind of problem. An oscilloscope comes in handy for finding this trouble (which looks like excessive ripple or the output of an unfiltered bridge rectifier), but with a simple voltmeter, the problem will appear only as an unusually low voltage, or one that varies depending on the load from the S-100 boards.

I've seen this problem many times, in old Imsai and Altair frames mostly, and each time, accompanied by the above mentioned symptoms. If you are using an old S-100 frame, you may want to check the capacitors in the power supply and replace them while you still have your sanity.

Next Time

More S-100 mini-reviews, including more on TurboDOS-compatible slave processor boards. If you'd like to see a mini-review of a certain S-100 product, please let me know, and I'll be happy to give it the once-over.

Of course, we'll also have more letters, including feedback, S-100 tips, and hints.

As always, I encourage reader feedback, and welcome questions, suggestions, comments, and ideas for future S-100 Bus columns. You can contact me c/o Micro C, on CIS 70150,102, Source TCH54, via GEnie mail at DJHARDY, or at the above address.
Poor Man's Networking:
Connecting Two CP/M Machines

Fancy new networks are showing up right and left, but they aren't for the price conscious or for CP/M users. This one is. With a serial port and a disk of software you too can connect two 8080/Z80 machines and make them talk. Doug tells us about his super-cheap home-brew networking scheme.

Like so many others, I’ve yearned for a multiuser or networked system, but I’ve never had the money. This article describes a package I developed to allow CP/M computers to be networked for almost nothing.

Networking?
A network lets computers share resources. For example, you could sit at one computer and access another system’s data files. You could run WordStar, Express, or dBASE on one computer and print the output on a printer connected to another.

Commercial networks usually have one or more computers (with hard disks) acting as “servers” or “masters,” and several other computers (“workstations” or “slaves”) sharing the masters’ resources.

Usually each slave and each master has to have a network interface board (costing between $300 and $500). These special interface boards often require coaxial cable to handle the high speed data transfers. All told, this kind of system costs big bucks.

I Can’t Do It On The Ritz
I own an Altair with a Compupro Interfacer 4 and a Big Board. Both have serial and parallel ports. I have access to an Apple II clone, a NorthStar Advantage, and a Kaypro II, all of which have serial ports.

The serial port is the common link. All of the above computers are capable of communicating at 19,200 baud, fast enough to handle a large amount of information in a reasonable amount of time. Even 9600 baud is usable, but below that, response time becomes a problem.

System Requirements
After fiddling with the requirements for a network, I came up with this list:
1. Must be cheap.
2. Must not involve buying any additional hardware.
3. Must not take too much memory.
4. Must work acceptably fast.
5. Must be cheap.
6. Must not require a hard disk.
7. Must allow sharing of printers and disk drives.
8. Must work with an unaltered CP/M 2.2.
9. Must work without interrupts and be easy to install.
10. Must be cheap.

Networking For Poor Folk
My Poor Man’s Network is simply a BIOS extension that’s loaded into memory just below the CCP. (Or you can move your CP/M down by 6K and load the network software above CP/M.)

It’s a generic BIOS extension, since it contains no disk parameter tables, just space for them. The disk specifications are loaded into the tables as they are received from the remote computer.

Capabilities
Poor Man’s Network allows you to read files from or write files to another computer, even though the systems may have different types of drives.

For example, one computer might have 8” drives, while the other has 5” drives. You use the remote’s disk drives by specifying the appropriate drive letter, as if it were your own drive.

Or you can share one or more drives with another computer on a Read Only basis. For example, you may allow the remote computer to run programs off your A: drive, while you retain read access to that drive.

Additionally, you can redirect output (LST; and TTY;) to another computer. This allows you to use the other computer’s printers or plotter. Or you can redirect input (RD:) from another computer, useful for sharing paper tape readers, digitizers, or other input devices.

You can also send one-line messages to the other computer.

A special driver section defines the actual interface to the port; there is nothing inherent in the package itself which implies the use of a particular kind of port. Nor is there a requirement for a hard disk, and no limitations are made on the number or types of disks.

What It Isn’t
As with all things in life, you don’t get anything without limitations. Following are a few of them:
— To simplify the programming, only two computers can reside on the network.
— Both computers must be running CP/M 2.2. Other versions of CP/M and other operating systems (including CP/M emulators on clones) are not yet supported.
— No alterations are made to CP/M, other than copying and replacing the BIOS vector table.
— 6K of memory is required, which may be located above CP/M (if you can move your CP/M down 6K) or in residence below the Console Command Processor (CCP), which itself takes 2K.
— Disks may not be removed while the network is active.
— Programs which bypass the normal CP/M calls to access the disk I/O directly can’t be used in a network environment. These include formatters, track by track copy programs,
some disk inspection/alteration programs, and disk test programs.

For example, if you run a copy program that directly accesses the disk controller on your machine, and you attempt to copy from your disk to a drive on the remote computer, you will probably wind up copying onto a disk on your own system.

- Data sharing is on a drive basis, not a file basis. That is, you give the remote user access to an entire drive. You may restrict access to Read Only or allow both Read and Write. If you give the other computer read access, you retain read access yourself. If you give read/write access, you lose all access to that drive until the network is terminated.

Using It

To start Poor Man’s Network, each user enters a SIGNON command which specifies how the drives and printers, etc., are to be shared.

For example, suppose we have a SuperKludge 80 with three 8” drives, and a KayPlunk II with two 5” drives. On the SuperKludge, you might enter

SIGNON B:R C:W H:

which means “my drive B: is available to the KayPlunk on a Read Only basis, my drive C: is available on a read/write basis, and I want a drive (H:) on this computer.”

On the KayPlunk, you might enter

SIGNON A:R F: G:

which means “my drive A: is available to the SuperKludge on a Read Only basis, and I want two drives (F: and G:).”

Poor Man’s Network on the SuperKludge will communicate with its alter ego on the KayPlunk and match up the drive requests. Thirty seconds after typing the SIGNON command, each computer will respond with “Network successfully established,” and the fun begins.

On the SuperKludge, you can do things with drives A: (which you kept), B: (which you are sharing), and H: (which is really the KayPlunk’s drive A). If you try to read from or write to your drive C:, you will get an “Access denied” error message, then a BDOS Select Error, and a warm boot. Since you gave write access to that drive to the KayPlunk, you no longer own it (at least until you type SIGNOFF or hit the reset button). The same thing will happen if you try to write on your drive B: or drive H:, because both of these are Read Only.

The KayPlunk has access to its own drive A: (Read Only), drive B: (Read/Write), drive F: (Read Only) and drive G: (Read/Write).

You can run any normal CP/M program as usual, specifying the drive letter for any of the drives to which you now have access. For example, on the SuperKludge, you can copy a file from the KayPlunk’s drive A: (your drive H:) to your drive A: by typing a PIP command —

PIP A:=H:GRONK.TXT

Or you can run programs from the KayPlunk’s disk A: by preceding the program name with H: —

H:STAT B:*.*

You can even make his disk your currently logged drive by entering —

H:

Amazing, isn’t it?

What’s more, there is no requirement that the disks be floppies. If you want to share a hard disk, or a RAM disk, it works just as well.

Sharing of printers, paper tape readers, etc., is done similarly; if you add the parameter LST:R to the SIGNON command, you’re saying “my printer is available to the remote”; if you just specify LST: then you’re saying “I want a printer.”

How It Works

Poor Man’s Network consists of three main modules — Setup, Debug, and NetBios, which are combined to make a single load module called SIGNON.COM.

Each module has a constant and an instruction part. The constant part contains all the variables and messages used by the module; the instruction

(continued next page)
(continued from page 75)

part contains all the machine instructions required for that module.

Setup gets control when SIGNON.COM is first executed. It analyzes
the parameters specified on the SIGNON command and complains if it
doesn't understand.

If the parameters are okay, it checks to see which disk drives are available
and then stores the disk parameter tables in NetBios. This means that
once the network is running you can't change disks. After all, changing a
disk would invalidate the disk tables.

Setup then relocates NetBios (and optionally Debug) to the top of memo-
ry. If you know how to create a CP/M system 6K smaller than normal, you
can tell Setup where to put NetBios by specifying the actual address in a
variable in the overlay file. Otherwise, Setup checks to see if any other BIOS
extensions are loaded, and relocates NetBios below the lowest extension or
below the CCP. If debugging is called (a parameter when starting SIGNON),
the code uses an additional .75K.

Setup now tries to talk to the other computer. It always politely listens
before talking. If you invoke the debugging option, you can watch this
process. If a minute passes without a response, setup displays "Failed to
communicate with remote!" and dumps you back into CP/M.

If communications are established, the systems exchange system IDs and
version numbers. (Thus, an upgraded version could support multiple comput-
ers.)

One system becomes the "master" and the other the "slave." The only
difference is that the slave will listen a little longer before talking. (Helps keep
the two systems from talking at once.)

Then the two computers exchange drive tables and Setup creates its vec-
tor tables. The CP/M BIOS vector table is copied into the relocated NetBios,
and the NetBios vectors are copied into CP/M. The address at location 6 is
altered to protect NetBios (if NetBios is located below the CCP), and then the
program balls back into CP/M with a warm boot.

Control
NetBios takes control every time a CP/M BIOS routine is called. That is,
every call to check console status, input or output a byte, read or write a
disk, or whatever, is intercepted by NetBios. This is invisible to users,
because NetBios maintains a type ahead queue to prevent keystrokes
from getting lost. Each time there's a call, NetBios checks to see if the
request is coming from the local key-
board or from the remote system.

Disk activity with CP/M consists of
several calls to BIOS routines to set
the track, set the memory address, set
the disk number, etc. All of these calls
are stored until there's a read or write
request. When the request arrives,
NetBios checks to see if the request is
local or remote.

If it's local, the operation is per-
formed immediately and the results
returned. If it's remote, then a request
record is built and transmitted to the
remote computer, which performs the
action, and returns a status record and
possibly a data sector.

Installation
Before Poor Man's Network can be
used, it must be installed with the
appropriate I/O drivers. Drivers are
provided on the distribution disk for
Big Board I and II; Kaypro II, 2X, 4,
and 10; Apple II with Super Serial
board; Televideo 803 and Portable I;
Compupro Interfacer 4; NorthStar Ho-

teron; and NorthStar Advantage. You
get both Assembler source and hex, so
you can edit and reassemble the
source driver if you wish.

You install the hex driver with DDT.
You can alter the screen parameters
and several other operational defaults
with DDT. If you have another kind of
computer, you will have to provide
your own driver, either by modifying
one of the drivers supplied, or by
writing one from scratch. (Note that
the drivers in MODEM7 are very simi-
lar to mine.)

Poor Man's Network is available for
$69 from me at the above address.

----

PUBLIC DOMAIN SOFTWARE
FOR *ALL* COMPUTERS FROM THE LARGEST LIBRARY IN THE WORLD!
Rent or buy complete user group libraries! (Or select individual disk: ONLY $4.00 each.)
Since it's not copyrighted, PD software may be freely distributed. Get accounting, dbase, Games, Word Processors, Educational,
Communications, Utilities, Spread Sheets and hundreds more! User Group libraries are expanding every month as new software is
contributed, get the latest info: a directory disk and catalog — only $5.00 postpaid! (Specify computer).
Browse through a rental library — copying out programs of interest to build your own very own special interest library!

<table>
<thead>
<tr>
<th>IBM-PC COMPUTERS</th>
<th>Rent</th>
<th>Buy</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM-PC Sig 154 Disk sides</td>
<td>$460</td>
<td>$845</td>
</tr>
<tr>
<td>IBM-PC Sig 154 ET Disk sides</td>
<td>$660</td>
<td>$1,215</td>
</tr>
<tr>
<td>PC-Blue 187 Disk sides</td>
<td>$190</td>
<td>$375</td>
</tr>
<tr>
<td>Capilto-PC 35 Disk sides</td>
<td>$40</td>
<td>$105</td>
</tr>
<tr>
<td>Author's Showcase 25 Disk sides</td>
<td>$50</td>
<td>$100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMMODORE 64</th>
<th>Rent</th>
<th>Buy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commodore 64</td>
<td>$69 from me at the above address</td>
<td></td>
</tr>
<tr>
<td>Commodore 128</td>
<td>$250</td>
<td>$500</td>
</tr>
<tr>
<td>Commodore 1532</td>
<td>$180</td>
<td>$360</td>
</tr>
<tr>
<td>Commodore 164</td>
<td>$200</td>
<td>$400</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>APPLIANCE 64</th>
<th>Rent</th>
<th>Buy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple 120 Disk sides</td>
<td>$100</td>
<td>$200</td>
</tr>
<tr>
<td>Amiga 25 Disk sides</td>
<td>$25</td>
<td>$50</td>
</tr>
<tr>
<td>Amiga 30 Disk sides</td>
<td>$50</td>
<td>$100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Macintosh UG</th>
<th>Rent</th>
<th>Buy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Macintosh UG</td>
<td>$250</td>
<td>$500</td>
</tr>
<tr>
<td>Macintosh UG</td>
<td>$300</td>
<td>$600</td>
</tr>
<tr>
<td>Macintosh UG</td>
<td>$400</td>
<td>$800</td>
</tr>
<tr>
<td>Macintosh UG</td>
<td>$800</td>
<td>$1,600</td>
</tr>
<tr>
<td>Macintosh UG</td>
<td>$1,200</td>
<td>$2,400</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CP/M COMPUTERS</th>
<th>Rent</th>
<th>Buy</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP/M UG 290 Disk sides</td>
<td>$75</td>
<td>$150</td>
</tr>
<tr>
<td>CP/M UG 92 Disk sides</td>
<td>$45</td>
<td>$90</td>
</tr>
<tr>
<td>CP/M UG 290 ET Disk sides</td>
<td>$150</td>
<td>$300</td>
</tr>
<tr>
<td>CP/M UG 290 ET Disk sides</td>
<td>$300</td>
<td>$600</td>
</tr>
<tr>
<td>CP/M UG 290 ET Disk sides</td>
<td>$600</td>
<td>$1,200</td>
</tr>
<tr>
<td>CP/M UG 290 ET Disk sides</td>
<td>$1,200</td>
<td>$2,400</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ATARI ST</th>
<th>Rent</th>
<th>Buy</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATARI ST</td>
<td>$125</td>
<td>$250</td>
</tr>
<tr>
<td>ATARI ST</td>
<td>$250</td>
<td>$500</td>
</tr>
<tr>
<td>ATARI ST</td>
<td>$500</td>
<td>$1,000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DISK SIZES</th>
<th>Rent</th>
<th>Buy</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD 25 Disk sides</td>
<td>$25</td>
<td>$50</td>
</tr>
<tr>
<td>STD 50 Disk sides</td>
<td>$50</td>
<td>$100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DISK SIZES</th>
<th>Rent</th>
<th>Buy</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD 125 Disk sides</td>
<td>$250</td>
<td>$500</td>
</tr>
<tr>
<td>STD 250 Disk sides</td>
<td>$500</td>
<td>$1,000</td>
</tr>
</tbody>
</table>

Call (818) 727-1015 for 3 minute info recording. Call (818) 941-0925 for technical info. Call (800) 621-5640, wait for tone, then dial
"PUBLIC" for orders only. Send large $3.95 stamped envelope for free catalog.

NATIONAL PUBLIC DOMAIN SOFTWARE LIBRARY, 1533-A AVOHILL, VISTA, CA 92084

76 MICRO CORNUCOPIA, #31, Aug-Sept 1986
XEROX® 820-1 & 2 Assembled & Tested
820-1 $50.00 820-2 $70.00
CP/M (8") 35.00 CP/M (8") 35.00
820-1 w/CP/M 70.00 820-2 w/CP/M 90.00

Terms: All orders are F.O.B. Hayward California - Prepayment or C.O.D. - California residents add sales tax - Prices subject to change without notice - Quantities limited to stock on hand

OEM HOBBYIST
Sips Inductors Pots 7400 Dips
Sockets Regulators Caps 74LS Chips
Linear Resistors Some ECL and more...

Authorized dealer for:

5 1/4" DSDD
80 Track 96 TPI
Philips®Drives
$100.00

Z80-B 6MHz $4.50

3P+N POWER SUPPLY
Single board power supply kit with 2 positive fixed voltages. +1 and -1 variable.
KIT $54.95

E²I COMPUTER PRODUCTS
2273 American Ave. #8 - Hayward, CA 94545
(415) 786-9203

“IF YOU'RE LEARNING C AND NOT USING ECO-C88, YOU'RE WORKING TOO HARD!”

Eco-C88 is a full C compiler for MS-DOS machines. Everything you need is included in the low price of $59.95, including a full-screen editor. Here’s what some reviewers are saying about the Eco-C88 C Compiler:

"Eco-C performed well on all the benchmarks, generating code that was quite comparable to that of compilers 10 times as costly." - Christopher Skelly, Computer Language, Feb., 1986

"This compiler does handle syntax errors much better than average - no avalanche of spurious messages here." - William Hunt, PC Tech Journal, Jan., 1986

"Eco-C88 is a high-quality package ... convenient to use ..." - Dr. David Clark, Byte, Jan., 1986

"Eco-C is definitely a bargain ... it includes both the compiler and an excellent Turbo style editor ... a useful compiler for advanced applications, and will serve far beyond the beginning phase." - Gary Entsminger, Micro Cornucopia, April-May, 1986

The compiler comes with a standard library of over 200 functions, and "mini-make" utilities, ANSI language enhancements (e.g., prototyping), expanded user’s manual, plus much more, all for the low price of only $59.95. Also ask about our support products!

If ordered with the compiler, the C library source code (excluding transcendentals) is $10.00 and the ISAM file handler (as published in the C Programmer's Library, Que Corp.) in OBI format is an additional $15.00. Please add $4.00 for shipping and handling. To order, call or write:

Ecosoft Inc.
6413 N. College Avenue
Indianapolis, IN 46220
(317) 255-6975 - 8:30-4:30
1-800-952-0472 (orders only)

Trademarks: Eco-C88, Microstat (Ecosoft), CP/M (Digital Research), MS-DOS (Microsoft), PC-DOS (IBM), Z80 (Zilog), 8086, 8087, 8088 (Intel).
A Cheap Megabyte For The Slicer

This is another one of the cheap upgrades. In fact, if there's any way to make a system faster and bigger and cheaper, we'll soon hear about it. Shawn wins cheap honors this issue with this 1 Meg RAM upgrade for the Slicer.

I wanted to bring up Concurrent CP/M 86 on my Slicer, but I knew from Laine Stump's column (86 World) that my 256K wouldn't provide enough elbow room for CCP/M 86. I needed more memory. I couldn't afford a megabyte expansion board ($800 then), so I decided I'd have to scheme my own extra RAM.

Laine mentioned a 512K upgrade in issue #26 of Micro C, dubbing it the "Thick Slice" (you've heard of the Fat Mac?). I dug in.

How much RAM? Well, 512K would give room for a 250K RAM disk and a few concurrent processes, but a full Meg sure had a nice ring to it. Besides, I really needed room for an additional process — to get my technophbic housemates up and running.

This article shows you how to upgrade to either 512K or a full megabyte.

The Trick

The existing Slicer RAM system is shown in Figure 1. Nice, clean, fast — 256k. But the irony of DRAM is that it's so reliable, yet forgets itself a few hundred times a second. So, enter the TMS 4500 RAM controller to remind the RAM. ("Hey RAM, remember that bit?")

The trick to this upgrade is to fool the 4500 into refreshing and controlling two or four times as much RAM.

This upgrade is possible because:
1. 256K chips are pin compatible with 64K devices. Pin 1 wasn't used on 64Ks, and is used as the 9th multiplexed address line (MA8) on 256K chips.
2. Only multiplexed address lines M0-M6 are used for the internal refresh address on the 256K chips (and most 64Ks, excluding TI and Micron Tech chips). So the extra multiplexed address line (MA8) isn't even involved in DRAM refreshing. Your old DRAM controller will continue to serve faithfully.

Now, since the DRAM refreshing is the same, we have to accomplish only two things:
1. Select the DRAM controller and support chips over the full range of RAM addresses.

RAM Select

The chip selecting problem would be most elegantly fixed with a software change (the 80186 has programmable memory select lines), but the maximum space programmable to a single select line is 256K, so that doesn't help. We'll have to do a hardware mod.

Refer to Figure 2. I latched the CPU's S2 signal with a spare portion of the LS373 latch at U12 (ALE triggered). Since S2 is essentially "memory select," NANDing it with UCS/ (ROM chip select not) gives us RAMCS/.

In summary: Just NAND UCS/ with a latched S2.
Multiplexing

My first (successful) multiplexing strategy was to let the CAS signal drive the multiplexer. However, CAS must not reach the RAM before MAB settles, so CAS has to be delayed.

Thanks to Mark York (CA), who has 512K up reliably, I found another approach. He suggested that since RAM latches the row address on the falling edge of RAS/, RAS/ can do the multiplexing too. By the time the addresses are multiplexed, the RAM will have latched the (old) row address.

So I used (RAS0/ NAND RAS1/) to drive the MUX select line of the LS157. The multiplexing takes about 25ns (from falling edge of RASx/ to MAB); RAM requires 20ns row address hold time (from falling edge of RAS). A bit tight, so DON'T use ALS parts!

Power Trips

Hey, before you get into the middle of this, make sure your power supply can handle the extra current required by the bigger RAMs. My data manual suggests that 256K chips use about twice as much current as the 64Ks, so your power supply must deliver about an extra amp (for the 1 Mbyte upgrade, half that for the 512K).

Details

My upgrade fits on a small daughterboard (see Figure 3), mounted on top of the Slicer at socket U12. You did use a socket at U12, didn’t you? Oh well, nothing a solder sucker can’t fix.

The daughterboard gets power and ground through U12, plus access to 3 addresses and latched S2. In addition, 5 signals (RAS0, RAS1, UCS, RAMCS, and MAB) must be hardwired to the board.

If there’s enough interest, I’ll build and sell the daughterboards (rough guess: $25 AT, $18 kit, $12 PC board). Please drop a line or post card if you’re interested, and I’ll get back to you.

For a megabyte, you’ll need double stacked sockets, or you’ll have to solder RAM chips piggyback, like in the DSI memory upgrade (Micro C, (continued next page)
issue 26, p. 51). Don’t bite off more than you can solder.

I used Fujitsu MB81256-15 RAM chips (150ns) at about $2.90 each. I bought them from MicroProcessors Unlimited (who gave excellent service and chip installation advice). (Editor’s note: be sure to get Japanese RAMs; U.S. 256K parts have been very poor quality.)

READ THIS: Buy plastic chips. The double stacked sockets won’t accommodate the somewhat larger ceramic chips, and the ceramics I received had metal tops, so the lower socketed chips shorted the upper ones.

Flight Restrictions

Any expansion board RAM will be a prime target for conflict, due to contending (offending?) memory selects.

I called Slicer for help determining restrictions for all those additional boards I don’t own. They were very helpful:

1. Those of you with the old 256K RAM expansion board may do the 512K main board mod. Jumper the expansion RAM to start at segment 8000. Then generate the main board RAMCS/ signal with: A19/ (latched) NAND S2 (latched). Note: grab A19 from U12 (pin 12) and invert it with a spare LS00 gate on the daughterboard to make A19/.

2. Those with the video board: video RAM starts at segment B000h, so you may do the 512K main board upgrade too. You’ll have to generate the RAMCS/ signal the same as the 256K RAM expansion folks above.

3. No, you megabyte expansion board folks can’t get another meg. You already have yours. Go away. (However, if you blow your own expansion board PROM, bank switch all the expansion board RAM, and devise your own RAMCS/ signal for the appropriate range, it might be possible to get 1.8 Meg. But don’t quote me.)

Here’s the small print (disclaimer time): as far as I know, the Slicer modifications I describe will work on all Slicer versions. I disclaim any warranties, expressed or implied ... squirm, squirm.

Theory To Reality

To begin, you’ll need: a weekend, a very small soldering iron (and a soft touch), RAM (16 or 32 pieces plus spares), and prototyping equipment.

I suggest you construct your daughterboard using rub-on transfers on a single sided copper PCB. Transfers, PCB, and etchant are under $10 at Radio Shack. Just follow the layout in Figure 3.

Voiding The Warranty

I found the RAM replacement and connection to be the most tedious part of this project. Double stacked sockets are awful to work with. No wonder they can’t get auto-insertion tooling to do it. I’d heard that the unions are too strong these days ... 

The biggest problem stems from the fact that the PC board was laid out with pin 1 tied to ground (for power bus convenience); remember pin 1 wasn’t used with the 64K chips. So, you’ll have to CAREFULLY bend every RAM chip’s pin 1 as shown in Figure 4. For the lower socketed chips, flatten pin 1 onto the chip’s back, then bend it north. For the upper chips, just bend straight up. (Editor’s note: you may only get one shot at the bending. The second time the leg breaks.)

I know, it makes you woozy to deface $120 in chips, but it’s the most painless way I can think of. If you’ve got a better idea (more respectful to the chips, nicer looking, etc.) let me know!

Construction

1. READ your RAM distributor’s static precautions. The smaller geometry necessary to develop the 256K chips also leaves them more susceptible to static.

2. Disconnect your Slicer board and get comfortable. Remove existing RAM; kiss good-bye.

3. Bend pin 1 for all RAM, as shown in Figure 4. If you have double stacked sockets, cut an eighth of an inch from the rest of the pins (see Slicer manual for help). Now install the first (only?) 512K in the lower sockets (U40-U55). Finally, solder all pin 1s together with a jumper wire (makes your Slicer look like it has braces).

4. Temporarily tie RAM pin 1 (MA8) to ground and power up to see a whopping 128K! That’s good! If RAM errors appear (oops), use the memory test (MT) portion of the monitor to isolate and replace the bad chip (you did buy a couple extra RAM chips, didn’t you?).

5. Full megabyte upgraders: Put the top set of RAM chips in, and again bus the pin 1s together. Make sure that upper RAM chip legs don’t short out on lower chip legs. Tie all pin 1s to ground, flame on, and see your 256K back up and running. Your Slicer is usable while you work on the next step.

6. Build the daughterboard, as shown in Figures 2 and 3. It will contain the new RAMCS/ and address multiplexing circuitry. Install a wire wrap socket in the center position, with solder tail sockets for the LS00 and LS157. Populate the daughterboard and strap the address configuration jumper for the 512K or 1 Meg option now. You expansion board users: build your additional mods into...
the daughterboard.

Note: I'll give signal origins below, but I found it easier to solder to feedthroughs to get signals to/from the daughterboard. Follow traces carefully, and do continuity checks.

7. 512K only: For 512K (permanent or temporary), you must replace A17 with A19 at the REN1 input (U25, pin 3). You'll find a convenient feedthrough between pins 10 and 11 of U25. Cut this trace and apply A19 (U12, pin 12). This mod will select the lower socketed bank over the first 512K.

8. Find S2 at J3 (pin29) and jumper it to the latch at U12 (pin 3). Prepare to install the daughterboard by soldering a jumper wire to the following signals:
   - RAS0/ U25, pin 6
   - RAS1/ U25, pin 7
   - UCS/ U6 (ROM), pin 20 (socket pin 20, not chip pin 20; the chip is smaller than the socket)
   - RAMCS/ U25, pin 4. On the solder side, follow the trace (toward the 80186) to the first feedthrough. Turn the board to the component side, grab the signal at the feedthrough, and cut the connecting trace (ouch).

10. Remove the chip at U12, install it in your daughterboard. Connect the RAS0/, RAS1/, UCS/, and RAMCS/ signals to the daughterboard. Jumper MA8 from the daughterboard to the RAM (pin 1). Remove the temporary jumper from MA8 to ground.

11. Plug the daughterboard into the U12 socket. Cross your fingers, meditate, pray, etc. Now power up and see your 512K or 992K (1 Meg minus ROM space).

12. Boot up, use the Setup program to configure yourself a 300K or 750K RAM disk. Reboot and copy your whole A: drive to the RAM disk. Be impressed. Oooo, tubular. That's a totally awesome RAM disk fer sure!

Debugging
Do you have the right signals? Check continuity and solder joints. Use the monitor and a scope to check signals RAMCS/ and MA8.

Short section, huh? I had zero problems, so I have no brilliant insight.

Wrap-up
Believe it or not, both my first PC boards (CAS MUX and RAS MUX methods) worked right first time! I haven't purchased CCP/M 86 yet, and although it's a bit ridiculous, I've configured myself a 750K RAM disk. Funny, I never thought I'd use that much, but of course I already have (though I sometimes run out of directory space before I run out of disk space).

Note: I wrote this article glitch free on my upgraded Slicer.

References
Slicer manual: DRAM installation.

Microprocessors Unlimited: DRAM notes.
Build Your Own
Inexpensive Logic Analyzer
Part 2, The Software

The crickets are chirping, the air conditioner is running (it gets HOT in Missouri) and, once again, Don is working. With computers. Hardware won't work without software (sometimes it doesn't work with it), so here's a bare bones program to drive last issue's logic analyzer.

To run the software you'll need a computer, display (orange, lime, any flavor), and the logic analyzer card. The program will run without the logic analyzer card; the data won't mean much (third dimension information perhaps), but it will run.

The program in Figure 1 is intended to be a base to build upon. You can add bells and whistles to configure the software to your needs. You can input the necessary configuration data, trigger the card, and display the data. Additional features might include file or printer dump, timing diagram display (if you have a color card or Hercules compatible graphics card), disassembler, etc.

The program will accept the trigger words, clock qualifiers, and clock edge data and transfer the information to the logic analyzer. The analyzer is then released to acquire a 1024 by 32 bit block of data. When the data is loaded, control is returned to the software driver. An out is provided in the program to allow the software to regain control if the analyzer card hangs. (Say, a trigger word doesn't occur.)

After the data is latched into the analyzer, the software driver transfers and displays it. Then, of course, the whole process can be repeated.

If you have any questions or problems, feel free to call before 11 p.m. Missouri time. I hope you have as much fun building and using the analyzer as I did designing it. See you at the SOG.
If Code = 1 {Don't Care}
Then Trig1 := Trig1+Answer
Else (0 or 1)
Begin
Trig1 := Trig1+Bit_val*Answer;
Dontcr1 := Dontcr1+Answer;
End;
LoopCnt := LoopCnt+1;
End;
LoopCnt := 0;
Answer := 1;
For LoopCnt := 0 to 7 Do
Begin
Read(Kbd,Tempin);
Write(Tempin,' ');
Val(Tempin,Bit_val,Code);
If LoopCnt = 0 Then Answer := Answer
Else Answer := Answer*2;
LoopCnt := LoopCnt+1;
End;
End;
LoopCnt := 0;
Answer := 1;
For LoopCnt := 0 to 7 Do
Begin
Read(Kbd,Tempin);
Write(Tempin,' ');
Val(Tempin,Bit_val,Code);
If LoopCnt = 0 Then Answer := Answer
Else Answer := Answer*2;
LoopCnt := LoopCnt+1;
End;
End;
LoopCnt := 0;
Answer := 1;
For LoopCnt := 0 to 7 Do
Begin
Read(Kbd,Tempin);
Write(Tempin,' ');
Val(Tempin,Bit_val,Code);
If LoopCnt = 0 Then Answer := Answer
Else Answer := Answer*2;
LoopCnt := LoopCnt+1;
End;
End;
Large Memory Space From A Small Memory Model

Small model, large model. There’s a lot of difference when you’re pricing a new C compiler. Fortunately you’re not necessarily restricted to 64K with the small model compiler. In fact, Ron brings us 32-bit, 640K, unsegmented addressing on an 8088! With a small model C!

In a previous issue I mentioned the hobbyist’s dilemma of whether to stick with a “small model” C in MS/PC-DOS or to spring for a high-priced compiler with far-this/long-that goodies and full 640K capabilities. Being sorely tempted by any chance to buy a 286 machine—nothing less—goodies and full I’ve been trying ever since to justify MS/PC-DOS the expenditure, and I just can’t. Not priced compiler with far-this/long-that an AT clone whirring on my desk, but economic.

like buying a garden. Still, my little business does need another compatible, but what on earth can I do with a large model C of course mostly to process words is month or so that I can’t do with the small? never written a C program that needed more than 64K worth of code. In a crannies. M utility became so convoluted that even its creator forgot all its with.

fun of C is its elegance, directness, and compactness, and when I start generating 30+K worth of code, things are getting tight-lipped and serious. On a few occasions I’ve fantasized about having more than 64K worth of data for monster arrays. I always find, however, that not all the work areas are being used at once, and a little pointer magic lets the data space serve double purpose.

More frustrating in the long run is my hankering after 32-bit pointers for swapping interrupt vectors and dabling with screen buffers, yet MS-DOS provides function calls to do those very jobs, and at Assembly language speeds, too. It would be fun, I admit, to “**ptr + +” from time to time across the CRT display, tossing characters and attributes as I go. However, is that millisecond of delight really worth $250? I doubt it.

The only real hangup comes from the heap. Even programmers of moderation find that databases tend to exceed 64K very quickly. Long databases accessed in memory require long pointers. With this convenient need in hand can the purchase of a new compiler be excused?

Moving Instead Of Pointing

Alas, no. You can get around the 64K heap limit if your small version of C offers you a utility to move runs of memory from one place to another within the entire 640K. With this “move far” call you possess extended peek and poke capabilities. (Well, that’s what my Toolworks C calls them. Those of you suffering from flashbacks to BASIC can substitute less painful names.)

If these two operations don’t explicitly come with your package and you have access to in-line code or to a linking assembler, they are trivial to write. Just load ES:SI and DS:DI with pointers to source and to destination, load CX with the length of the run, and do a REP MOVSB. The 8086 family iterated move statement is unbelievably fast.

To illustrate the relative ease of substituting long moving for long pointing with small C, I’d like to create a binary tree that can exploit the full 640K. I’ll assume you know what a binary tree is and how effortlessly new application routines can be written once the basic tree has been designed.

Editor’s note: For those of you who weren’t around when the first computers were created, the binary tree was the first source of fresh bits (of both types). There were whole fields of binary trees (bit fields) and their bits were stored in cores (if memory serves me correctly). Now, of course, they’ve synthesized bits, so we’re turned to using binary trees for searching and other sorted affairs.

The operations shown in Figure 1 are vacuous, but once you see how to do it, no doubt you’ll think of uses.

Bargain Basement Far Pointers

To store addresses in the left and right branches of every member, we will need a 32-bit data type. I’ll use the long — redefined as a “farpointer” — as a pigeonhole, with the segment address as the high word and the offset as the low. Those of you who have very, very small Cs without the long data type can define a two-integer structure to do the same job, though the code will become a little more cluttered.

Every time the heap pointer (another long masquerading as a “farpointer”) is incremented, I’ll adjust the high and low words with modulo 16 arithmetic to avoid segment boundary problems. Once that’s done, to all events and purposes the heap exists in unsegmented memory addressed by a 32-bit pointer.

The heap will begin at the first paragraph of free memory, which is returned in the AX register by MS-DOS function call 48H. We will need to make that function call anyhow to
The traditional way to build trees is to set up a routine like so:

```
if (pointer null) {
    add member at top of heap
    return address
    use equals to set right or left pointer of previous node to address
} else if (test yields less than)
    set left member equal to next function call
else
    set right member equal to next function call
```

Consider, for example, the usual trick of the address of database branches to be beaten, but it's used to that.

Since binary trees are built (surprise!) on binary comparisons, we'll need two buffers within the data segment: one for the primary database element in question, and one for the element with which it's being compared. To add to the database, just poke the element from the primary buffer up to the top of the heap and push the heap pointer up by one unit's length.

To alter the pointers within a node, just peek the member down into the secondary buffer, change the right or left farpointer, and poke it right back where it came from.

The only other complication that comes with this shuffling is that the "equals" sign won't work as it does when honest pointing is going on. Consider, for example, the usual trick of using the recursive function return of the address of database branches to set the right or left pointers of nodes. The traditional way to build trees is to set up a routine like so:

```
if (pointer null) {
    add member at top of heap
    return address
    use equals to set right or left pointer of previous node to address
} else if (test yields less than)
    set left member equal to next function call
else
    set right member equal to next function call
```

Elegance in action. Unfortunately, through the magic of high-speed shuffling, the various elements all have the same working location, the secondary buffer inside the data segment. Any recursive function call returns a heap address of a branch to that branch itself. That bit of circularity makes for strange binary trees indeed.

What one must do instead is keep track of (a) a global storage area containing the 32-bit address of the previous node examined and (b) a flag telling whether the branching there was from the right or the left. So long as one is hopping merrily from node to node, no changes need be made. But when an unoccupied branch is finally reached, the secondary buffer must have its resident right or left farpointer altered, and the node must then be poked back up where it belongs.

A lesser complication occurs on all recursive searches. To point to an element is in fact to load it into a buffer. Hence the buffer contents change both as one climbs out upon and as one returns back along the various branches. The traditional binary tree operation —

```
if (pointer not null) {
    do it to element pointed to by left pointer
    do it to element pointed to by right pointer
}
```

must therefore be modified to —

```
if (pointer not null) {
    load element
    do it to element pointed to by left pointer
    reload element
    do it to element pointed to by right pointer
}
```

**How Much Slower?**

The printem() function in Figure 1 illustrates this twist. All this sleight of hand sounds complicated and, worse than that, slow. But it's really not. The efficiencies of recursion are still available to you, and it's fast enough with the 8086 string move calls that you'll never notice the difference, since binary searches in trees involve a minimal number of recursive calls.

I tried to measure the overhead of the method by doing repeated simple linear searches through the entire 2500 member database from beginning to end. Granted, it's a gross misuse of the tree structure, but it does measure the "pointing" overhead rather directly. The tree generated below took about 3.4 seconds to run through. For comparison I created an identical tree in Turbo Pascal, which uses long pointers directly, and the trek took about 0.8 second.

Since no practical use of binary trees I can think of spends more than one percent of the CPU time doing searches, who cares if finding a member takes forty milliseconds rather than ten? The important things are access and capacity, and those we have in abundance.

**Complications**

As I said, those of you patient enough to follow me this far have had at least a tad of experience with binary trees and can see the method and adapt it to practical situations. If you don't mind working with headers and doubly linked lists, you could go all the way and simulate the standard Unix-style malloc() and free() functions, though I find it easier to flag a database member I don't want to "see" and get rid of it permanently while rewriting back to disk. The space saved not having to deal with allocation headers more than makes up for the occasional hole in the heap: just stack 'em up like pancakes.

Figure 1 shows how I create the database with a random number routine and then list the integers to be sure the tree was properly formed.

Once you see how the tree is created, examining and modifying database elements is easy. You may still decide to buy a "long" C compiler anyhow, but you won't be able to use database operations as your excuse.

**Figure 1 - Binary Tree Program**

on pages 86 and 87.

---

**Micro Cornucopia #31, Aug-Sept 1986**
#define MAX 2500 /* a sufficiently large number */
#define LEFT -1
#define RIGHT 1
#define ROOT 0
#define NIL 0x0L /* to steal a term from Pascal. Just */
#define farptr long /* a simple "0" won't do */
#include "stdio.h"
#include "regs.h" /* header to define structure to hold
register storage */
struct record{
    char stuff[126]; /* the primary database element */
    int t;
    farptr left,right;
}prim_buff,sec_buff;
struct pointer{
    int _ofs,_seg; /* a convenience for overlaying longs */
};
farptr rootptr,heaptr,last,alloc_far();
int size;
struct regs rr; /* register storage */

main()
main()
    {int i;
    size=sizeof(struct record);
    rr.bx=0x7000;
    rr.ax=0x4800; /* Dear MSDOS, may I have 70000H bytes to play with? */
    interrupt(0x21,&rr); /* MSDOS function call */
    heaptr=0x10000*rr.ax; /* initial segment address to high byte */
    for(rootptr=NIL,i=1;i<=TOP;i++){
        prim_buff.left=prim_buff.right=NIL; /* the actual tree formation */
        prim_buff.t=rand(); /* Here's where you'd get practical */
        treemake(rootptr,ROOT);
    }
    printem(rootptr); /* trivial test to see if it worked */
    }

/*treemake(next,hand)*/
treemake(next,hand)
    {int hand;
    
        if(next == NIL ){
            if(hand==LEFT) sec_buff.left=heaptr;
            else if(hand==RIGHT) sec_buff.right=heaptr;
            else rootptr=last=heaptr; /* special treatment for root */
            unyank(&sec_buff,last,size); /* last node sent back home */
            unyank(&prim_buff,alloc_far(size),size); /* new member to the top of */
                /* the heap */
            }
        else{
            yank(&sec_buff,next,size); /* try the node */
            last=next; /* store address of last node */
            if(prim_buff.t<sec_buff.t) treemake(sec_buff.left,LEFT);
            else if(prim_buff.t>sec_buff.t) treemake(sec_buff.right,RIGHT);
        }
    }

MICRO CORNUCOPIA, #31, Aug-Sept 1986
printem(pp)
    farptr pp;
{
    if(pp != NIL){
        yank(&prim_buff,pp,size); /* see text for explanation of the second "yank" */
        printem(prim_buff.left);
        yank(&prim_buff,pp,size);
        printf("%d\n",prim_buff.t);
        printem(prim_buff.right);
    }
}

/***********/
yank(place,ptr,n)
    char *place;
    farptr ptr;
    int n;
{
    /* In my implementation, "place" is the */
    peek(seg(ptr),ofs(ptr),place,n); /* buffer address inside data segment. */
    /* This would be a nice place for inline code. */
}

/***********/
unyank(place,ptr,n)
    char *place;
    farptr ptr;
    int n;
{
    poke(seg(ptr),ofs(ptr),place,n);
}

/***********/
farptr alloc_far(len)
    int len;
{
    farptr bottom;
    int inc;
    /* normalizing addresses with modulo 0x10 work */
    bottom=heapptr;
    /* shifts and and's are just faster */
    inc = ofs(heapptr) + len;
    heapptr = ( seg(heapptr) + (inc >> 4) )*0x10000 + (inc & 0xf);
    return bottom;
}

***************/
int seg(fp)
    farptr fp;
{
    struct pointer *pp;

    pp = &fp;
    return pp->_seg; /* return high byte of long */
}

***************/
int ofs(fp)
    farptr fp;
{
    struct pointer *pp;

    pp = &fp;
    return pp->_ofs; /* return low byte of long */

End of Binary Tree Program
The Green Hills C Compiler

Full Optimization For The 32032

There are three commercial compilers available for the DSI-32 board, all of them from Green Hills: Pascal, FORTRAN, and C. Of the three, the one most used by the Definicon staff is the C compiler. Dave covers the strengths of this implementation of the language.

One of the compilers available for the Definicon DSI-32 board is the Green Hills C compiler. This single pass, fully optimizing compiler uses advanced techniques and is a full implementation of the C language, as defined in "The C Programming Language" by K&R in addition to a particular implementation of C, a few timers are bound to flare. Some C compilers implement K&R, but leave out bit fields. Others don't have floating point. But the Green Hills compiler has all the language features. In addition, the Definicon MS-DOS interface brings features of the Unix implementation of C to the DSI-32.

The Preprocessor

The Green Hills compiler also includes another feature of the Unix C compiler — the preprocessor. In fact, since the Green Hills compiler is a single pass compiler, the preprocessor is actually part of it, rather than a separate program. This speeds up compile times for large programs.

The preprocessor has two reserved variables: __FILE__ and __LINE__. __FILE__ contains the current filename of the program being compiled, and __LINE__ contains the integer representation of the current line number. During debugging, the following code could be used:

```c
if (a > b) printf("Warning: line \%d in \%s, a>b\n", __LINE__, __FILE__);
```

which would print an error message if 'a' was greater than 'b,' containing the current filename and line number.

Enumerated Data Types

Enumerated data types are supported as well, allowing you to define your own type of variable. For example, if you were writing a traffic control program, you might wish to have a variable contain the current traffic light color. You can do this as follows:

```c
define color {red, yellow, green} light;
```

This creates two items: an enumeration identifier (color), and an enumeration variable (light). To make the light red, we simply assign red to the variable light:

```c
light = red;
```

Since we have a new enumerated type defined, we can declare another "color" variable easily:

```c
define color light2;
```

Note that since the color identifier is already associated with the values red, yellow, and green, we don't need to reiterate them. The variable light2 may now be used as light is in the previous example. As you can see, the enum keyword operates in much the same way as the struct keyword.

More Features Of Green Hills C

Bit fields are often not implemented in C compilers. Not so in the Green Hills compiler! To declare a bit field in a structure, follow the variable name with a colon, and the length (in bits) of the bit field:

```c
define BITS

  int active:1;
  int ready:1;
  int count:4;
  int last:2;

struct tape;
```

This declares the structure tape. To use BITS, treat it as a normal structure reference:

```c
tape.active = 1;
tape.ready = 0;
tape.count = 7;
tape.last = 0;
```

In the Green Hills compiler, a bit field may have a maximum length equal to the least amount of storage possible.

Bit fields can be used to save data space. Imagine a program that records Yes or No responses to a number of questions. By using an array of structures with bit fields, the amount of memory (and disk space) used for each set of questions can be reduced significantly.

Bit fields are also used in graphics. On the IBM Color Graphics adapter, either one or two bits are used in graphics modes. By having an array of structures of bit fields, you can represent the entire bit image of the graphics adapter in your C program.

The ASM Function

Another feature of the Green Hills compiler is the ASM function. This function permits the use of assembly language directly from C, as follows:

```c
main()

  printf("Hello, world!");
  asm("movq 0,r0");
  asm("movd 15,r0");
  asm("svc");
```

On the DSI-32, this would cause a "terminate" service call. The ASM function allows code to be written to take advantage of machine instructions, not normally generated by C.
MICRO CORNUCOPIA'S C CONTEST

If you've been going so hard you just can't C straight, then drop everything, dig out the old compiler, brush off the cobwebs, and C what you can C.

Any kind of C. From Small-C to large, from a hundred lines up, this is your chance to C'cure fame and glory (plus valuable prizes).

GRAND PRIZE:
• DSI-32, 1 meg, 10 Mhz 82032 computer board with assembler/linker from Definicon Systems
   21042 Vintage St., Chatsworth, CA 91311
• Optimizing C Compiler (large model)
   from Manx Software, Box 55, Shrewsbury, NJ 07701
• Essential Software's C Library
   (graphics, I/O, everything)
   from Essential Software, Inc
   Box 1003, Maplewood, NJ 07040
• 3 year subscription to Micro C

ENTRY FORM

Program Title _______________________________________________

Purpose ___________________________________________________

Which C compiler? __________________________ Target system(s) __________________

I hereby release this program to the public domain and give Micro Cornucopia the right to print the listing.

Signature ________________________________________________

Name _____________________________________________________

Address ___________________________________________________

City __________________ State ________ Zip ________________

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

--- CONTEST DEADLINE November 1, 1986 ---
Little Board™ .... $249
The World's Least Expensive CP/M Engine

**CP/M 2.2 INCLUDED**

- 4 MHz Z80A CPU, 64K RAM, Z80A CTC, A-32K EPROM
- Mini/Micro Floppy Controller (1-4 Drives, Single/Double Density, 1-2 sided 40/80 track)
- 2 58332C Serial Ports (75-9600 baud & 75-38, 400 baud), 1 Centronics Printer Port
- Power Requirement: +5VDC at .75A; +12VDC at .05A (1-4 Drives, 2 RS232C)
- Only 5.75 x 7.75 inches, mounts directly to a 5-1/4 drive
- Comprehensive Software Included:
  - Enhanced CP/M 2.2 operating system with ZCPR
  - Read/write/format dozens of floppy formats (IBM PC-DOS, KAPPRO, OSBORNE, MORROW...)
  - Menu-based system customization
  - Operator-friendly MENU shell

**BOOKSHELF™ Series 700**
Fast, Compact, High Quality, Easy-to-use CP/M System

**Priced from $895.00**
10MB System
Only $1645.00

- Ready-to-use professional CP/M computer system
- Works with any RS232C ASCII terminal (not included)
- Network available
- Compact 7.3 x 6.5 x 10.5 inches, 12.5 pounds, all-metal construction
- Powerful and Versatile
- Based on Little Board single-board computer
- Entire system mounts into single slot drive
- 10-mB internal hard disk drive option
- Comprehensive Software Included:
  - Enhanced CP/M operating system with ZCPR
  - Word processing, spreadsheet, relational database, spelling checker, and data encrypt/decrypt (T/MAKER III*)
  - Operator-friendly MENU shell, Menu, Friendly*
  - Read/write and format dozens of floppy formats (IBM PC-DOS, KAPPRO, OSBORNE, MORROW...)
  - Menu-based system customization

**DISTRIBUTORS**

1.2 Meg Floppies On XTs

Bob Carol called me one afternoon and mentioned that Newport Components of Newport Beach CA was selling a special 1.2 Meg drive and support software to run on PCs and XTs. Well, at 1.2 Meg per disk, you have a decent way of backing up winchesters. At $190, the price seemed reasonable.

I called them and asked if I could borrow one to review. "Sure," they said.

Well, it came. The software worked okay; the drive, on the other hand, didn't.

The drive looked like a cheap version of a standard floppy. It connected to a standard XT floppy controller which meant that, unlike the AT drives (which transfer data at 500K bits/sec), it was moving data bits at 250K/sec. So the spindle had to turn at half the rate of the normal 1.2 Meg AT drive.

That's a pretty heavy handicap. The intensity of signal is proportional to the velocity of the disk under the head. That's why marginal drives usually have problems on the inner tracks.

I checked out the system. Sure enough, it formatted and wrote, but when it came time to verify the data, it choked. On the inner tracks. And it did it on super quality 1.2 Meg certified, unfolded, very expensive disks.

I returned the drive. I still have the disks. They work great on our 286i (but not at all on standard DD or QD systems). Oh well.

Meanwhile, the word is that MS-DOS 3.20 ($75) comes with a driver for quad density drives. You get 720K per

EDITORIAL

I've spent years perfecting my working space. My #2 Big Board has been allowed to remain because it's very, very quiet (and very, very heavy).

When I work on a PC I often use personal noise reducers: ear plugs and earphones. They help, especially if the music is right, but that's not the whole answer.

The Kaypro 2000 is very quiet (it doesn't have a fan) so I purchased a base unit for it. Unfortunately the base unit has a nasty, uncouth blade spinner that runs constantly, even when the 2000 is turned off.

I'm sure there are quiet fans. We're talking about moving air here, not calling out the volunteer fire department.

ARC Enemy

There's reportedly a version of ARC (fancy MS-DOS library utility often used in conjunction with bulletin boards) that erases the system tracks on hard drives. (Heavy duty worm ware.)

The latest valid version of ARC is version 5.12. The worm ware version identifies itself as 5.13, but there's no way to get it to identify itself without exposing your system tracks to its malevolence.

However, the good version of ARC.COM (which splits itself up into ARC.EXE and ARC.DOC) is 59K. The worm version is 34K (and doesn't split). The copy on the Micro C bulletin board works fine.

262-2262, TEL 041-6364
disk and you can use all those generic, off the shelf, under the counter, and over the wall disks that have been just itching to hold data again.

Cheap Clones

I’ve been watching prices of clones and clone boards lately. The figures I’ve seen lately in PC, PC World, Computer Shopper, and Byte have been about double the prices listed by Taiwanese manufacturers. In other words, a $695 system in Byte (a small empire) would cost about $350 in Taiwan (also a small empire).

However, there have been predictions that minimum level 4.77MHz clones will be available for $300 before Christmas (here in the U.S.). It’ll be interesting to see if it happens.

I know, for a fact, that you’ll be seeing some excellent prices on ATs and 80186 equivalents of ATs. In fact, board prices for these units will soon be approaching the levels I’ve seen lately in Taiwan (also a small empire). The figures I’ve seen lately for ATs and 80186 equivalents of ATs.

Anyway, in addition to their regular (very competitive) price sheet they also include a sheet of specials. In their price sheet they also include a sheet of specials. In their advertisement has some interesting prices. I’ve called them and they sound real, but I haven’t ordered from them so I can’t attest to their service.

May Computers
8210 Katella Ave #D
Stanton, CA 90680
(714) 897-2037

I’m still happy with the service we’ve been getting from Sky High. The only complaint I’ve heard about them is that communications on the phone can be a bit of a struggle (it helps if you put it down on paper and mail it). If you get the wrong thing, they’ll exchange it. See the Sky High ad in this issue.

Also, I just saw an ad for a 1200 baud, standalone, Hayes compatible, almost lookalike, modem for $129. (When the Duck finds out, feathers will fly.) Unfortunately I misplaced the $129 ad. Oh well, by the time I find it there’ll be a $99 ad for me to lose.

(Wait a minute, I remember where it is... Somewhere. Here. BCE Liquidation is selling a General DataCom AcuLine 1200. It’s a 1200 baud, standalone, Hayes compatible with software for $129. 15 day trial, 2 year warranty etc. etc. etc. I haven’t dealt with them, so you’re pioneering on this one. Let me know how you make it.)

BCE Liquidation
3233 K St NW
Washington, DC 20007
(800) 545-7447

(continued next page)
EDITORIAL

(continued from page 91)

Reflections On Mirror

Last issue I mentioned Mirror, the $49.95 super communications package. I also mentioned that it was slow writing to disk. Well, it turns out that it opens and closes the file each time it writes. It can’t leave its file open because it might be running in the background (and with open files spotted about, things get very messy for MS-DOS). This is definitely my favorite modem package, and at $49.95...

Anyway, a number of you called to find out how to contact SoftKlone (the Mirror folks).

SoftKlone
(800) 538-8157
(800) 676-3470 CA

Advertisers

All of this talk of packages and cheap boards brings us to advertisers. The good ones are great. I love reading the ads in Micro C, Byte, and Computer Shopper. I love reading good catalogs. Hey, children don’t have a monopoly on the wish books.

On the other hand, the bad advertisers are a pain. I mean, you’re all ready for Christmas (whenever it is) and you get junk, or nothing at all, or they didn’t tell you the most important part.

“Oh, it doesn’t work? But it will — all you have to do is replace the drive controller board.” (Which in our case meant also replacing the serial port, parallel port, and real-time clock.)

The technician on the other end of the line wasn’t upset. He didn’t have to be; I was upset enough for both of us. Putting a tape backup on a clone should be trivial, but I’d already spent two months trying to figure out why their software wouldn’t select their drive. MS-DOS didn’t have any trouble.

“Oh, of course, I still can’t guarantee that it’ll work. We haven’t completely figured out when it will work and when it won’t.”

They didn’t say that up front. When Larry placed the order.

“Hey, our tape drives are turkeys, but if you want to piddle around with one, and try to make it work — then we’ll agree to send one. Eventually.”

They didn’t say that. But they should have.

Fortunately, I didn’t see their ad in Micro C. In fact, I asked them if they’d ever heard of Micro C. They hadn’t.

“Good,” I said. “It’s a bad place to advertise.”

If you’d like to know who I’m talking about, it’s Micro Design International in Winter Park, FL. It turns out that they are selling the Alloy APT-40, a 40 Meg tape drive manufactured by Peripheral Technology. (Peripheral Technology also wrote the buggy software.) I’d avoid anything with an Alloy drive in it for the time being.

Amiga Watch

I’ve been assuming that Amiga and the ST are thrashing it out, blow by blow. I’ve been figuring that if
the Amiga goes down it'll be at the hands of Atari (or vice versa).

But I might be wrong. As long as the Amiga remains a home machine (a majority of the sales so far) its real competition is the Commodore 64 and 128. (You've heard of the Commodore 64. It's the only computer slow enough to run alongside an Apple II without snickering.)

Insiders are attributing most of Commodore's $36 million loss last quarter to the Amiga. In fact, word is that Commodore has cancelled several large Amiga software projects because of poor hardware sales.

Commodore had a 2-month $500 off special on the basic system with color monitor ($1295 for the package) in an attempt to get systems off dealers' shelves and into the hands of folks who might purchase software (so there'd be more software sales and more hardware sales and more software and more hardware...).

Company officials noted that they weren't going to be continuing the special because they couldn't live on $1295. Dealers, on the other hand, weren't sure they could make a living selling a games machine for $1795.

Of first quarter industry software sales, 3.4 percent were for the Amiga, 24 percent were for the 64 and 128 (source Micro Marketworld). Atari ST software accounted for a mere 1.5 percent.

Finally

Insiders are telling interesting stories. Commodore is planning to sell its IBM clones in the U.S. (they've been doing well in Europe). And, the company may be trying to kill the Amiga by simple neglect.

The long-awaited software package that lets the Amiga emulate the PC is now available, but it suffers the same problems as the SWP 8088 board. It doesn't support sound or graphics. Commodore has published a list of the programs (and versions of programs) that do run, but the list isn't very long. Also, PC programs take 2 to 4 times longer to run on the Amiga than they take on the PC.

Editors of Commodore mags feel that the Amiga would have to be priced at $595, complete, to become a significant player in the home market. And even then it has a problem because there isn't an upgrade path from the 128 to the Amiga.

Perhaps Commodore should have come out with a 64 emulator rather than a PC emulator. That way no one would have noticed how slowly it ran.

Meanwhile, Commodore can count on continuing sales of the 64 and 128 (continuing at least until PC clones become cheaper than the 128, which shouldn't happen before, say, September).

I don't think I'd buy any Commodore stock right now.

***

<table>
<thead>
<tr>
<th>Pick A Few Of Our Freshest Disks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MS-DOS Disks — 5½” or 3½”</strong></td>
</tr>
<tr>
<td><strong>NEW MS-24 Superior Games</strong></td>
</tr>
<tr>
<td>Core Wars, SPACE WAR, Dungeons &amp; Dragons, and TRONLC. (TRONLC includes Source.)</td>
</tr>
<tr>
<td><strong>NEW MS-25 Ultra Utilities</strong></td>
</tr>
<tr>
<td>U-ZAP, U-FORMAT, and U-FILE: Map diskettes, unerase files, format non-standard disks, interrogate sectors &amp; much, much more!</td>
</tr>
<tr>
<td><strong>NEW MS-26 Expert System Generator &amp; Tutorial</strong></td>
</tr>
<tr>
<td>Introductory expert system builder (ESIE), example expert systems, &amp; detailed tutorial.</td>
</tr>
<tr>
<td><strong>NEW MS-27 System Primer</strong></td>
</tr>
<tr>
<td>Device drivers and utility tutorials with source and DIAGS - the super system snooper.</td>
</tr>
</tbody>
</table>

— 23 more MS-DOS disks in our catalog —

| **KAYPRO CP/M Disks - 5½”** |
| **NEW K-44 Pascal Runoff - Printer Utilities** |
| Print sideways, on both sides, or design your own character font. Source included. |
| **NEW K-45 Pascal Runoff - Utilities** |
| File manipulation and encryption, sorting routines, and much more with source |

— 43 more KAYPRO CP/M disks in our catalog —

**ALL DISKS $8.00 each**

Check the catalog for 8” CP/M disks. Also for Kaypro ROMS, Schematics, and Add-ons.

Our catalog is FREE. Ask for one today.

Technical Calls: 9 am - noon, M-F 503-382-8048

Micro Cornucopia | P.O. Box 223 Bend, OR 97709

9 - 5, Weekdays Order 503-382-5060
**Digital dynamics**

- **NEW** - SEETIME has alarms! You can set up to 4 alarms, each containing date, time & message.
- **NEW** - Screen saver feature turns off the screen after a set time with no keyboard activity.
- **NEW** - Configuration program to set SEETIME parameters:
  - time/date location on screen
  - military or civilian display format
  - cursor blinking or steady
  - screen saver timeout

**SEETIME** can input the time or date into any program, print the screen, display alarms.

**SEETIME** is part of TIMEPRO:
- APPT personal schedule manager
- TD file time/date stamping & backup
- LOG computer usage logging

**TIMEPRO** runs on '84 & later Kaypros with Kaypro, Kenmore or Advent clocks.

$49.95

**SEETIME upgrade for TIMEPRO owners** ... $14.95

Shipping and Handling - $2.50

(206) 772-0291  Visa, MasterCharge
12448 83rd Ave S., Seattle WA 98178

---

**FORTRAN**

*a structured language? Here's How!*

New RF-77...a Ratfor-like preprocessor for PC FORTRAN

- WHILE, FOR and REPEAT loops.
- Free form input. No column 1, 6, & 7 to worry about.
- DEFINED constants and INCLUDED files.
- Outputs standard ASCII source code. May be compiled by any Fortran 77 compiler.
- Runs on any MS-DOS/PC-DOS equipped personal computer.

$65. Postage paid. Texas residents add $3.98 tax.

Logical Developments
P.O. Box 55798, Houston, Tx 77255
For VISA/MasterCard orders call: 1-800-835-2246, ext. 41

---

**MODULAR DISTRIBUTED DATA ACQUISITION SYSTEM**

**AD1208**
DATA ACQUISITION SUBSYSTEM

- 8 single ended channels
- 12 bits plus sign resolution
- Input range $+4.096V$ standard, $-4.096V$ optional
- User installable input attenuator option for each channel
- 15 samples/sec sampling rate
- Interfaces to any host via RS-232 port 300 to 19200 baud
- Stand alone (requires RS-232 to modular jack adapter) or multisystem (connects to MP8000 multiplexer) operation
- Simple one or two character commands from any language (BASIC, PASCAL, ASSEMBLY etc)
- Programming examples included
- Modular interface cable included

Assembled and tested ............................................ $199

25 pin RS-232 to modular jack adapter kit (specify male or female) ..................................................... $10.95

**CN100**
RS232 MULTIPLEXER CONTROLLER

- Connects to host via standard RS-232 cable
- 300 to 19200 baud operation
- Controls up to 16 MP8000 multiplexer boards
- Simple two character command from any language (BASIC, PASCAL, ASSEMBLY etc)
- Requires at least one MP8000 multiplexer board for complete RS-232 multiplexer

Assembled and tested ............................................ $115

**MP8000**
RS232 MULTIPLEXER

- 8 channel RS-232 multiplexer expansion board
- LED numerical display for selected channel
- Easy connection to CN100 via 20 wire ribbon cable expansion bus
- Expandable up to 128 serial channels with one CN100 controller

Assembled and tested ............................................ $125

NOTE: CN100 and MP8000 are designed for family of STELCOR Data Acquisition and Control subsystems, but may be used as a general purpose RS-232 multiplexer for applications not requiring an active handshake signals.

STELCOR
P.O. Box 51362
PALO ALTO, CA 94303
(415) 988-0558

4 channel analog output plus 16 digital I/O subsystem coming soon

**TERMS:** Check or Money Order. California residents add 6.25% sales tax. Add $5 for shipment outside USA. Allow 2 weeks for delivery.
Technical Torture

Across

1 LIFO storage
6 Language whose lines are numbered
11 Hackers' projects
16 Lowest form of communication
18 Short for GA, GA
19 Someone who can sell IBMs at list
20 RS-232 "may I send" line
22 Load instruction
23 Post script
24 Two mice
27 Greek harp
29 Fe, ... Fo, Fum
31 4-bit broad
36 Address Latch Enable
38 Government authorized lady
39 CP/M command
40 Cat talk
41 Kingdom by the ______
42 Complement of Write Only Memory
43 Slimy vegetable
44 Emergency room
45 Compound statement delimiter
46 French Freud potatoes
47 Herbert ________, AI man
49 Short version of 26 Down
50 Cat talk
51 Logoff
52 Field location after five doubles
57 Not all
58 80186
60 Where I's and O's grow
62 Florida Technological University
63 Of the fourth degree
64 Officers Training School
65 Irrational ratio
66 Expensive papers
67 Code, Art, or Amp
68 Bit set value
69 Drop this in Adventure to get bird
70 If it ain't on, it's ______
72 Smile again
74 Low power Schottky
75 Grumpy computerist
81 Souffle staple
82 Baby bringer
84 Repeat while equal
85 Talking horses
86 White Mountain State
87 Sews shorts
89 Chernobyl heat source
91 One of the Kettles
92 Magnum
94 4B4C
95 Lithium
96 Major distress
98 Going between
101 Taiwanese ROM
104 Acceptable
105 Serial input/output
106 Famous July celebrations

Down

1 Micro
2 The late S-100 columnist
3 What thespians do
4 IC tests
5 Keep it stupid, simple
6 To ____ or not to ____
7 A boolean oxymoron
8 Peru's continent
9 ____ _______ 's Way
10 Center of Gravity
11 Hitler's party
12 For example
13 Ties the knot
14 Adventurer's foe
15 Access charges
17 Old Testament sequel
21 Samarium
23 Hot memories
25 Positively charged carrier
26 Long version of 49 Across
28 If at first you don't succeed ... 29 Hard to negotiate; harder to collect
30 Soft helmet
32 Hot memories
33 Predecessor of PROLOG
34 Chrome _________
35 Famous root beer
37 Fine linen fabric
44 _________ while
46 Itty Bitty Machine Co.
48 Boatman's boolean
50 Chicken with its vowel removed
51 Leave out
53 Extra Special Transistors
55 French loot
56 What you never have enough of
59 Four Tandons
60 Binary player
61 Circuit on a chip
62 Memory savings and loan
65 Programmable friends
66 Businessman's lunch
71 Ultimate nerd
73 One-armed serial chip
76 His machine beat Adlai
77 Wrath
78 Wire service
79 French article
80 Measurement of pressure
83 Logic gate
88 First In First Out
90 After Lying Stand Up
91 Fast port
92 16th Greek letter
93 ______ and outs
97 Snake sound
98 Rhode Island
99 A pebble game
100 Rhenium
103 Prefix for two

MICRO CORNUCOPIA, #31, Aug-Sept 1986 95
Statistics And Warfare, 
A Little Calculated Fun

With all those really useful languages around, is anyone still using BASIC? Yep. And they’ve written a statistical (not sadistical) package with it. Not a bad one.

Plus, you’ve no doubt heard about all those fun people who create worms (those nasties that zap screens and disks). Well, now you too can be a fun person by creating worms that wipe out your opponent’s worms.

If you’re looking for a first-rate, inexpensive statistical package for your PC, try Number Cruncher Statistical System. Despite its mouth-garbling title and low price ($79.95), it’s a full-featured, easy-to-use, speedy system.

NCSS (for short) was written in Compiled BASIC (someone has been taking BASIC seriously) over the last six years (I tested version 4.21). It requires 196K of memory and allows you to analyze up to 250 variables and 32000 observations.

Correlation/Regression Analysis — bivariate, correlation analysis, multiple regression, interactive variable selection, robust regression, discriminate and principal component analyses;

Analysis of Variance — t-tests, one-way-, twoway-, and nway- ANOVA, repeated measure;

Time Series Analysis;

Nonparametric Statistics — sign test, Wald-Wolfowitz runs, Wilcoxon matched pairs test, Mann-Whitney two sample tests, Kruskal-Wallis test, Friedman’s block/treatment test, and nonparametric correlations.

In addition, you can graph data in box and scatter plots.

The NCSS database is flexible. You can sort, back up, transpose, and delete it. You can merge or join it with other databases.
other databases, and you can even copy individual columns between databases if you need to.

NCSS includes a fullscreen editor for viewing and modifying databases and an excellent command-by-command tutorial.

The package isn’t flashy — a small spiral-bound notebook and disks — but it’s outstanding for the price ($79.95).

For more info contact —

Dr. Jerry L. Hintze
865 East 400 North
Kaysville UT 84037
(801) 546-0445

Core Wars

No doubt some of you are spending too much of your time “working” before your terminals (and monitors). If so, take a break, and exercise your mind and your computer’s memory with Core Wars (available on the Micro C bulletin board and on MS-DOS disk 24).

Each of two players writes a program in a low-level Assembly-language-like language. These two programs are loaded into a circular area of memory (actually an array of several thousand locations whose last address is contiguous to the first), and executed alternately, one instruction at a time.

The object of the game is to wipe out your opponent’s program. Whoever’s program stops executing first loses.

Nine instructions —

```
    MOV A B
    ADD A B
    SUB A B
    JMP A
    JNZ A B
    JNZ A B
    CMP A B
    DAT A B
    ```

and three addressing modes —

- Direct (default)
- Indirect ("#")
- Immediate ("#""")

are your weapons.

For example, the simplest battle program is —

```
    MOV 0 1
```

which copies the current location to the next, and then advances to the next location and repeats itself, writing over everything in its path. Eventually, if unchecked, it will write over the opposing program and win the CORE WAR.

Sound crazy (and fun)? It is. Several sample programs on the BB (and MS-24) will get you started.

Happy battling, and may all your wars occur in cores.

For further reading —

MS-DOS Express 2.0

Thank your Aunt Molly (or Uncle Alfred) that high-quality software at least sometimes comes at affordable prices. Take MS-DOS Express, for example, a full-screen editor worth twice its price ($39.95).

Express 2.0 is the commercial version of Laine and Cecil Stump’s speedy little public domain editor (Express 1.0). It includes access to DOS functions (DIR, TYPE, etc.), an excellent macro-key facility, very fast search and replace, and flashy variable-speed, bi-directional scrolling. It’s configurable, so you can make it look like WordStar.

In order to see how Express stacked up to more expensive editors, I decided to abandon my favorite (VEDIT Plus) for one issue and use Express exclusively. Happily, I made it through, missing VEDIT Plus only occasionally.

Express doesn’t have all the features of WordStar or VEDIT Plus, but it has most of them and it’s faster. For example, it loaded a 15K file in 6.5 seconds to VEDIT’s 16.5 and WordStar’s 15.0. And it saved a 15K file and exited to DOS in 10.0. VEDIT Plus performed the same trick in 12.0 seconds, WordStar in 19.0.

In sum, Express is an impressive editor, and in conjunction with ROFF4 (a text formatter available in the public domain) will fulfill most of your editing needs.

For more info —

TCI
17733 205th Ave NE
Woodinville WA 98072

20 Megabytes For The X16

If you’re thinking about adding a hard disk to your X16 (PC Tech’s 80186 version of the clone), you might benefit from Dick Simpson’s experience. He writes —

“You gave me the name of two suppliers, and I hit paydirt with PRO-TO, Inc. My call to them was answered by Mary Jo, an enthusiastic, helpful person who knows all about the X16. I bought a Seagate 225 20 Meg hard disk for $350 and an Ompti 5100 controller for $168, probably the best prices around. The Ompti is a true controller and the fastest one supported by the X16B ROM. And it runs with an interleave of 1:1! This thing is fast.

“By the way, the IBM XT type enclosure wasn’t designed to take a SCSI controller (as PC Tech points out). But I was able to fabricate a very satisfactory mount.

“I cut 2 pieces of 1/4 inch aluminum rod 11 7/8 inches long, then drilled and tapped each end for 6-32 screws.

“I then drilled holes through the rods to clear 4-40 mounting screws and mounted them over the drives so that the 34-pin header on the controller card was directly over the 34-pin connector on the hard drive (the 20-pin connector is offset with this method, but you can easily manipulate it).

“This gives you a stable and neat installation. The 50 trace cable furnished with the controller is long enough to go from the motherboard, across the cabinet, and up over (with a 90 degree bend) and into the header on the controller.”

Thanks, Dick.

For more info —

PC Tech
P.O. Box 128
904 North 6th St
Lake City MN 55041
(612) 345-4555

Proto PC
2439 Franklin Ave
St. Paul MN 55114
(612) 644-4660

And that’s Tidbytes (oops, I mean bits).

```
BIOS Patch
This program allows Turbo Pascal V3.0 to work with Z80 systems where the BIOS uses IX and IY without restoring them. The patch is installed at location 0040H (this space is reserved by Digital Research for patching CBOS). I use the program so Turbo will work on a Bondwell 2 lap portable.

John Newgas Turbo Pascal BIOS Patch Shown at Right

Tip For Microline Users
Microline 92 printers are available with standard and IBM Plug-N-Play software. By burning both versions into Q4 and Q5 and using the lower and upper half of a 27128, you can enjoy the best features of either one.

To do this, toggle pin 26 (A13) of Q4 and Q5 to 5V and ground. Simply cut the 5V traces between pins 26 and 27 of Q4 and Q5 (on the back of the board). This will isolate pin 26 of both chips.

Connect pin 26 of Q4 to pin 26 of Q5. Also connect a 1K pullup between pin 26 of Q4 and 5V (pin 28 of Q4).

Finally, connect a short (8") piece of wire to pin 26 of Q4 (solder it to the resistor). Connect another piece (use 2 conductor ribbon wire) to ground (pin 14 of Q4). Insert the new 27128 EPROMs.

Add a miniature switch to the end of the wires and mount it in the slotted hole available in the back of the printer. Use a large washer here. You may prefer to drill a mounting hole in a removable cover plate instead. There is just enough clearance to re-insert the board.

Now by simply turning the printer off and throwing the switch, either version can be booted up.

Joseph Malik
221 Bradley Dr.
Wilmington NC 28403

PC Reset Switch
A good hardware reset is invaluable when you’re thrashing around in low memory on Personal Clones. Simply shorting the power supply’s PWRGOOD line to ground should do the trick and probably will if your machine is a True Blue PC. The clones are another story. Many boards don’t look at PWRGOOD so we need to go elsewhere. Pin 11 on the clock generator is the reset line. Shorting this line to ground through a momentary contact switch (see figure) will guarantee a reset on all systems. Look for the clock generator (it’s an 8284) by the power connectors on the rear right corner of the main board.

Micro C Staff
LEAP INTO A NEW DIMENSION

RP/M2™ creates
CP/M®2.2 compatible
IBM PC

1. Remove the 8088
2. Install the NEC µPD70108 (V20)
3. Boot PC RP/M2

The µPD70108 chip is a fast 8088 that also executes 8080 machine code. PC RP/M2 is an operating system. Either standalone or with MSDOS present, PC RP/M2 provides the solid base of a genuine operating system reliably distinct from the facade created by an MSDOS interface. All 2.2 features are supported with 56.5k TPA, plus file date and time stamping, fast virtual disk, iobyte redirection, terminal cursor positioning, color console display. DOSDISK accesses MSDOS drives; MODEMPC transfers files via serial port; SETDISK redefines a drive to any of over 80 formats, allowing PC RP/M2 to directly process existing CP/M files in their native form. Two system disks plus 105 page user and technical manual $129. Shipping $5 ($10 nonUS)

AZTEC CII FOR CP/M $199
(Special price for Micro C subscribers $149)
MANX

Also available for Apple DOS, HDOS, CP/M-86, PC-DOS

MODULAR DATA ACQUISITION SYSTEM

We Bring Engineers, Scientists And The IBM-PC Together. Our unique high-performance modular data acquisition system allows you to purchase the configuration that exactly meets your requirements. All of Qua Tech's data acquisition products include LABSTAR, powerful software designed to give BASIC programmers total control of their application.

8 BIT A/D SYSTEM
12 BIT A/D SYSTEM

QUA TECH, INC.
478 E. Exchange St. Akron, OH 44304
CALL NOW (216) 434-3154

MICRO CORNUCOPIA, #31, Aug-Sept 1986 99
The following folks are reaching you for only $500.

**For Sale:**

- Slicer
- Postage. National postage. Datalife Head Oeaning System, Reg. $11.95 and cards accepted (preferred). Downloading-disk format Drive Cleaning Kit Sale!
- Flippies, $15.
- After receipt with 3 more Pascal-Z UG on 46
- Modified. Keyboard $25. Documentation (21 pgs.)/cable
- Public Domain ASCil; pad, caps-lock, repeat, self-test! Brand new, hundreds
- NJ sold to builders of Apples, Big Boards, Xerox 820s. Parallel
- 100 PC-Blue
- 706 209 9-5. National
- 20
- IBM-PC
- 9-5. National
- 1-30
- ROMS (4re)~ CPU/ROM
- 209
- VOL 1-175 MS-DOS, ,$185, directory disks $5
- 20
- 95CXiO Vol 1-246 on 17
- 8" IBM-PC
- 5.25" formats. Rental is for 7 days
- failures over the last three years.
- Power electronically switched. Can't
- toolbox and test
- into and test
- disk
- Turbo Pascal, dBASE, assemblers, utilities, communications or business applications, the Gold Mine offers the most current selection in the western U.S. Regular subscriptions are $25 per year. Contact The Lost Dutchman’s Gold Mine, P.O. Box 2957, Phoenix, AZ 85063, or call (602) 247-2880 to download an application. Money Back Guarantee! 31

Flight Plan Program with NAV-AID database covering entire U.S. “Install” for your airplane’s performance. CP/M or IBM. Supports user entered “custom” waypoints. Completely menu driven. Thoroughly documented. $29.95. For program description send SASE to NAVI-GATE, 3051 Shirley Dr., Newbury Park, CA 91302. 32

CP/M Emulates PC-DOS! Innovative utilities let CP/M reassign drives, autofind files (even overlaid), transform user areas into “disk drives,” create smart batch jobs, “window” the BDOS, format text columns, and much more. Inexpensive, copy-embled, 30-day trial. LOGIC ASSOCIATES, 1430 Thome, Chicago, IL 60660, (312) 274-0531. Ask for free catalog, review. 31

For Sale: IBM PC Clone (Silent Partner PC). 640KB memory, clock with battery, monochrome display, Hitachi color monitor. Genoa Specturm graphics adapter, two 5.25" floppy disk drives, Microcience 20Meg hard drive, Microsoft mouse, two RS232 ports, two parallel ports. Too much software to list. $2100 incl. freight. Kevin Tyrrell, 1221 Adobe Dr, SE, Grand Rapids, MI 49506. (616) 241-1902. 31

**New MBASIC CP/M Game Disk for video-able Kaypros (not pre-84 models), featuring:** Tank - Battle the enemy; Club Casino - Try your hand with the “One-armed Bandit”; Spacewar 1.1 - Blast the alien fiends; Tic-Tac-Toe, Hangman, Sketch Pad, and more. Send $8 for disk or SASE for information to: Eric Macy, Glenburn Star Rg., Fall River Mills, CA 90021. 31

---

**WANT ADS**

The following folks are reaching you for only 20 cents per word. If you would like to reach the same audience, send your words and 20 cents for each to Micro Cornucopia.

- **$25** Keyboards for computer builders - 83 keys, full ASCII; upper/lower case, all control characters, numeric pad, caps-lock, repeat, self-test! Brand new, hundreds sold to builders of Apples, Big Rigs, Xerox 80s. Parallel output, positive TTL logic, strobe. Uses only 100mA of +5 volts. Custom case available. 90 day warranty unmodified. $25. Documentation (21 pgs.) cabable
- **For Sale:**
- **Computer Doctor -** Will repair your SICK computer, drive, etc. Specializing in CP/M, Radio Shack, and IBM machines. Can also program EPROMS. SASE for more info. (914) 243-7160. Computer Doctor, Rt. 42, Box 190, Jackson, MO 63755, 31K

**For Sale:**

- **Computer Doctor** - Will send you the neatest “gadget” for only $12 postpaid, I will send you the neatest “gadget”
- **New MBASIC CP/M Game Disk for video-able Kaypros** (not pre-84 models), featuring: Tank - Battle the enemy; Club Casino - Try your hand with the “One-armed Bandit”; Spacewar 1.1 - Blast the alien fiends; Tic-Tac-Toe, Hangman, Sketch Pad, and more. Send $8 for disk or SASE for information to: Eric Macy, Glenburn Star Rg., Fall River Mills, CA 90021. 31

---

**WANT ADS**

The following folks are reaching you for only 20 cents per word. If you would like to reach the same audience, send your words and 20 cents for each to Micro Cornucopia.
NOW AT THE SBC MART
COMPUTING SALE-A-THON

THE BEST OF BOTH WORLDS
Run your CP/M programs on any IBM PC or compatible. RUN/CPM transforms your PC's floppy drives into CP/M drives able to directly read, write and format over 100 CP/M disks. RUN/CPM directly executes your CP/M programs. Kaypro, Morrow, Osborne, Cromemco, Heath, many more. Because this is a hardware solution, your programs run up to 30% faster.

* RUN/CPM only $79 (Requires NEC V20 chip below)

PC SPEEDUP KITS
* NEC's V20 chip will increase the throughput of your system up to 90%. That's almost double your present processing speed.

ADD-ON CARDS
- multifunction card 384K, clk/cal w/battery, serial, parallel, and game ports, with 8K save $50 Model MFC-4 $135
- monochrome graphics card runs 1-3-3 graphics, w/printer port, 720x348, MGC-1 now $135 (Hercules equiv.)
- floppy controller for 1-4 48tpi or 96tpi drives, w/cable save $25 Model FDC-2 now only $74
- color adapter w/light pen port, RGB and composite outputs reg $140 Model CC-1 now just $116

* better than the Super 7: floppy disk controller, clk/calendar, serial, parallel & game ports, spoiler & RAMDISK s/w Sale $145 ideal mate for 640K motherboards
- clock/calendar card reg $58, CL-1 $149
- OMTI hard disk controller card (this card is super fast) reg $220, Model HDC-1 now $175, w/cables

SBC PLEDGE
* service after sale * technical support
* low prices
* one year warranty

MOTHERBOARDS
- full IBM PC/XT compatibility
- 8 I/O slots
- runs IBM's PC-DOS 3.1
- BASIC interpreter available
- great foundation for business or personal system
- 1 year warranty

* Model MB-1 256K motherboard with 8K regularly $295, now $199
* Model MB-3 640K motherboard with 8K regularly $399, now $250
* Model MB-2 640K motherboard, 4.77 and 7+ MHz clock, w/256K Reg $399, now $297, this is a fast board.

ABOUT OUR MOTHERBOARDS
- the most IBM PC compatible motherboards available
- each runs all commercially available software
- each works with all commercially available add-on cards
- low cost
- one year warranty
- standard cables for 4 disk drives

HARD DISKS
- complete internal 1/2-high 10MB kit, incl. controller and cables, special $398.
- complete 10MB kit with low power 3-1/2" half-high drive, fits 5-1/4" slots, $469.
- 3-1/2" 10MB Winchester drive alone HD-3 just $380.
- 20MB half-high drive HD-2 now $395.

HARDWARE SPECIALS
- keyboard 5150 style regularly $125, Model KB-3 only $85
- keyboard 5151 style regularly $175, Model KB-3 only $99
- memory nine 150ns DRAMs Set ID-256 256K reg $55, now $34
- no better DS/DD drives anywhere, 48tpi F-4 $99; F-9 96tpi $129
- Smartam modem (fully Hayes equivalent) 300/1200 baud auto ans/dial, reg $399, now just $249

POWER SUPPLY
- 135 Watts
- side switch
- top quality
- one year warranty
- standard cables for 4 disk drives

MODEL PS-135 power supply regularly $129, now only $85

LAST MINUTE SPECIALS
* RUN/CPM only $79--see details in upper left corner of this ad
* IBM PC-DOS 3.1 #95 (only $65 with motherboard); Ellis PASCAL, COBOL, FORTRAN, BASIC compilers only $34 ea; NS mouse w/PC Paintbrush $145

* COMPLETE 100% IBM PC equiv syst #987

* 96tpi drive package: over 800X on a single floppy, complete kit with 96tpi drive, software #149

Shipping and handling: drives/motherboards/cases/power supplies $4.50 ea, cards $2 ea, software $2.50 ea, keyboards/modems $4 ea, speedup kits $2 ea, memory $1/ea, COMPUTERFACTS $3 first set then $1/ea. CA residents add 6% sales tax.

To order or for further information: CALL (619) 375-5744
The SBC MART, P.O. Box 1296, Ridgecrest, CA 93555
The SBC Mart is part of Computing Technology, 821-B Commercial Street, Ridgecrest, CA 93555

AMERICAN EXPRESS
TURN YOUR KAYPRO ON TO HANDYMAN!!

WHAT THE CRITICS SAID!

"...I've been saved! My faith has been restored in CP/M and my trusty Kaypro...performs as advertised...it's the best thing that's happened to Kaypro owners since we bought the computers in the first place. It all works fast, easily, and without a hitch, and it's got a 1001 uses! (at least). The documentation is excellent...what a blessing it can be. You start to use it and wonder how you lived without it."

Peter McWilliams
The McWilliams Letter
January 1986

"Doubles the usefulness of the computer. Installation is easier than opening a can of peas."

Alice Petersen
Genealogy Computer Support Group

"One of my favorite tools."

Ted Silveira

HANDYMAN is a plug-in board for your Kaypro!!!
HANDYMAN Works with all Kaypro CP/M models!!!
HANDYMAN Easy to install!!!
HANDYMAN We'll talk you through it over the phone!!!

HANDYMAN Includes software in ROM and its own working RAM!!!
HANDYMAN You never have to load it!!!
HANDYMAN Single stroke command!!!
HANDYMAN is compatible with all CP/M application programs!!!

HANDYMAN Uses no main memory!!!

NEW — HIGH TECH K-20 COMPUTER — INTRODUCTORY PRICE — $1,595.00

Includes:
All brand new components mounted in a Kaypro 10 case.
20 Megabyte Seagate 65 milliseconds hard-disk drive.
50 Kybte (quad-density) Panasonic floppy-disk drive. (Reads single and double sided double-density disks too.)
84 series Kaypro main board with modem and real-time clock.
Standard Kaypro software package (WordStar, etc.)
Advent Turbo ROM or MICROCode Kplus ROM. And of course HANDYMAN.

These are complete computers — not upgrades. Price and availability subject to change. Allow 3 weeks for delivery. Add 3% for credit card purchases (computers only) and $20.00 shipping and handling. For prices on 30, 40, 50, & 100 Megabyte computers. 90 day warranty.

NEW — HANDYMAN Version 1.3
Now complete with File Manager. Copy files from any drive/user to any other drive/user without leaving your original application file.
Now enter Appointments with a single keystroke.
Now available with Advent Turbo ROM or MICROCode KPLUS ROM. (Call)
And much more. HANDYMAN Still uses No Main Memory.

Still only $124.95

HIGH TECH RESEARCH
1135 PINE STREET, #107 — REDDING, CALIFORNIA 96001
CALL 1-800-446-3220 — IN CALIFORNIA CALL 1-800-446-3223

YES! Send me ___________ HANDYMAN(s) at $124.95 ___________
My Kaypro signs-on (boots) as CP/M 2.2 ___________ (letter).
(Required for 84 series Kaypros only).
Yes! Send me ___________ HIGH TECH K-20(s) at $1,595.00 ___________
Add 3% for credit card purchases (K-20s only)
California residents add 6% sales tax ___________
Shipping and handling ($5.00 HANDYMAN — $20 K-20) ___________
Signature ___________ TOTAL $___________

My Kaypro is a ___________ (model) ___________ (model year).
I use ___________ (Wordstar or Perfect Writer).
Charge my ___________
Card # ___________
Exp. ___________

C.O.D. and checks drawn on US banks also accepted.
Phone (____) — ___________ Eve (____) — ___________

Trademarks Kaypro (Kaypro Corporation) (MicroPro International Corporation) Perfect Writer (Perfect Software, Inc.)
XT CLONE COMPLETE SYSTEMS
MOTHERBOARD WITH 640K INSTALLED (TURBO)
150 WATT POWER SUPPLY  FLIP TOP CASE
AT STYLE KEYBOARD  COLOR GRAPhICS VIDEO
SERIAL PORT  PARALLEL PORT  REAL TIME CLOCK
FLOPPY DISK CONTROLLER
TWO FLOPPY DISKS  $750.00

ABOVE WITH ONE FLOPPY AND 20 MEG HARD DRIVE  $1175.00
TESTED AND ASSEMBLED WITH ONE YEAR GUARANTEE

KEYBOARD (MICROSWITCH) 113 KEYS 8 BIT ASCII
PARALLEL OUTPUT WITH CASE AND COMPLETE DOCUMENTATION  $40.00

SA800-2 DISK DRIVE  CLEAN - USED  $59.00
SA850 DISK DRIVE  CLEAN - USED  $99.00
STANDARD MOUNTING - GUARANTEED 60 DAYS

CABINET FOR ABOVE HOLDS TWO DRIVES
POWER WIRING COMPLETED (AC AND DC)
FAN AND INPUT AC WIRING DONE
20 X 9 1/2 X 17 DEEP
ALL MOUNTING HARDWARE

POWER SUPPLY FOR TWO DRIVES (SWITCHING)
WILL POWER DRIVES AND SBC 6 X 8 X 2 1/2
5VDC @ 8 AMP, +12VDC, -12VDC, -5VDC @ .5 AMPS EA AND 24VDC @ 2 AMPs
CABINET AND POWER SUPPLY TOGETHER  $100.00

ORDER CABINET WITH POWER SUPPLY AND
2 SA850 DRIVE FOR MICROC READERS  $250.00

RODIMe 20 MEG HARD DRIVE  R202E
55MSEC 4 HEADS 640 CYLINDERS  $350.00

XT CLONE PARTS ONE YEAR GUARANTEE
MOTHERBOARD TURBO (4.77 AND 8.00 MHZ)  $145.00
MONOCHROME GRAPHICS CARD WITH PRINTER $ 75.00
COLOR GRAPHICS CARD  $ 65.00
FLOPPY DISK CONTROLLER  $ 40.00
HARD DISK CONTROLLER  $125.00
DISK I/O CARD - FLOPPY DISK CONTROLLER
RS-232 CLOCK PARALLEL GAME ADAPTER  $ 95.00
CASE FLIP TOP STYLE  $ 45.00
POWER SUPPLY 150 WATT  $ 75.00
KEYBOARD AT STYLE  $ 65.00

QUME 142 HALF-HEIGHT FLOPPY DRIVE
IBM COMPATIBLE (LIMITED SUPPLY)  $ 89.00

CASCADE ELECTRONICS
ROUTE 1  BOX 8
RANDOLPH MN 55065
507-645-7997
FREE SHIPPING ON PREPAID ORDERS
COD ADD $3 CREDIT CARD ADD 5%  MN ADD 6%
LIMITED TO STOCK ON HAND  EVENING CALL OK

ADVERTISER'S INDEX
Advanced Concepts Engineering ....... 50
Ampro Computers ................... 90, 91
Anderson Techno-Products, Inc ....... 77
Andratech ......................... 92
Atlantic Publishing Corp ...... 23
Austin Codeworks ................. 67
BD Software ........................ 59
Bakkers, Andy ..................... 38
Borland International Inside Front Cover
Carolina Engineering Labs .......... 69
Cascade Electronics ............... 103
C Contest .......................... 89
C.C. Software ..................... 8
Computer Surplus Store .......... 39
Computview ....................... 29
Data Disk ......................... 37
Definicon .......................... 94
Digital Dynamics .................. 18
Digital Research ................... 49, 56
EcoSoft ............................. 77
Epic Sales ........................ 28
ERAC Co. ........................... 10
EZ1 Computer ...................... 77
Fletcher, Don ...................... 22
Halted Specialties ................. 1
Hawthorne Technology .............. 68
High Tech .......................... 102
Innovations ........................ 69
Integrand .......................... 27
Kenmore Computer Technologies ... 96
KYSO Technology ................... 75
Laboratory Microsystems .......... 51
Logical Development ................ 94
Logitech, Inc ....................... 11
McTek (Sky High Co.) ............... 36
Manx Software ..................... 99
Microcomputer Systems Consultants 73
Micro Cornucopia ................... 93
Micro Methods ..................... 99
Micromint .......................... 93
Micro Motion ...................... 100
Microprocessors Unlimited ....... 51
Microsphere ....................... 19
Micro Supply ....................... 21
Micro Technical Designs .......... 35
Miken Optical Co. ................. 67
Mitek ............................... 73
National Public Domain Software ... 76
Oliver Computers .................... 64
PC Tech ............................ 64
PC Tech . Inside Back Cover
Periphco ......................... 100
Poor Person Software ............ 51
Qua Tech ........................... 99
SBC Mart .......................... 101
Second City Software ............. 57
Servo Computers ................... 96
Slicer .............................. 81
SLR Systems ....................... 18
Soft Advances ..................... 47
Source Information ............... 47
Stelcor Inc. ....................... 94
Tomorrow's Computing ............. 63
Unified Software ................... 5
United Products ................... 46
Viking C Systems ................. 47
Western Wares .................... 38

MICRO CORNUCOPIA, #31, Aug-Sept 1986 103
In Search Of The Truly Random


What we’re really talking about when we say “random number” is a random series of numbers. Which leads one to ask, “When is a series random?”

For example, is

1,8,5,3,7,9,2,4,5,3,10

random in a range of 1 to 10?

Is

11,13,12,13,15,11,11,15

random in a range between 11 and 15?

Testing A Random Generator

You can determine the probability that a series is random with a chi-square test, which compares the expected with the observed number of occurrences. This comparison is calculated in the formula

\[ \chi^2 = \sum_{i=1}^{N} \frac{(O_i - E_i)^2}{E_i} \]

and then compared to a chi-square table of values to determine the probability of randomness.

In my two example sequences, the first (1,8,5,3,...) has about a 50 percent chance of having been generated randomly. The second has about a 5 percent chance.

Generating a truly random series is tricky. So let’s start by writing a “pseudo-random” generator.

Random In Pascal

We can create a pseudo-random series generator in Turbo Pascal using the linear congruential method (it’s a mouthful, I admit, but simple)

```pascal
function Random(Var R: integer):real;
begin
  Random := S MOD R + 1;
  S := (125 * S + 1) MOD 4096;
  {new S for next call to Random}
end; {Random}
```

We call the generator (function Random) with R equal to the highest number we want to allow in the random series.

A pseudo-random series will be good enough for most purposes (simple games, for example), but let’s try to do better. Fortunately, there’s an interesting way out of the depths. But first, let’s digress long enough to introduce a PROLOG version of Random.

Random In PROLOG

As in Pascal’s Random, we pass R (the range delimiter) to Random. The number generated for our random series will be passed back to our main program as N.

```prolog
Seed(S), /* "" implies "if"*/
random(R,N):- /* "":-"" means "if"*/
  Seed(S), /* "":-"" means "if"*/
  Random(R,N),
  N = (125 * S + 1) MOD 4096,
  asserta(Seed(NewSeed)).
```

The “predicate” Random is true if Seed(S) and each subsequent clause within the body of Random is true.

The two unusual predicates (retract and asserta) delete the old seed, S, and add a new seed, NewSeed, to a database built into PROLOG. The next time we call Random, Seed will be passed the new seed, S.

Seeding With The System Clock

But Random needs help if we’re going to generate truly random sequences. A solution — access the system clock and let time seed the generator.

The clock count is kept in low memory at address 0000:046C on the PC. You can watch it tick away easily with DEBUG —

```prolog
random(R,N):- /* "":-"" means "if"*/
  Seed(S), /* "":-"" means "if"*/
  Random(R,N),
  N = (125 * S + 1) MOD 4096,
  asserta(Seed(NewSeed)).
```

We can even randomly reseed Random to edge closer to the truly random by recalling the bios interrupt randomly (as often as we want in CX, a value unpredictable by the user).

By Way Of Closing

A question — does computer generated randomness have anything to do with artificial intelligence?
Features:
- 8 MHz 80186 microprocessor with true 16-bit data bus.
- True PC compatibility with our own custom ROM BIOS and PC card slots for the video of your choice.
- Standard 512K zero wait DRAM, or 1M options.
- 8087 numeric coprocessor support on board.
- On-board floppy disk controller for up to four 5¼" drives.
- On-board SCSI hard disk controller port. Supports WD1002 SHD, WD1003 ACS, OMTI 5100, 5300 ADAPTEC ACB 4000.
- 2 On-Board Serial Ports.
- Battery Back-Up Real Time Clock.

Designed, Manufactured and Serviced by:

PC TECH
P.O. Box 128
904 North 6th Street
Lake City, MN 55041
(612) 345-4555

COD, VISA, MASTER CARD, CERTIFIED FUNDS
OUR NEW PC 8700 AT-STYLE KEYBOARD AND BORLAND'S SUPERKEY® SOFTWARE

THE PC 8700 AT-STYLE KEYBOARD INCORPORATES EVERY IMPROVEMENT IBM DESIGNED INTO THEIR ADVANCED "AT" KEYBOARD. QUALITY SO GOOD, IT'S BACKED BY AN UNPRECEDENTED TWO-YEAR WARRANTY!

Advanced design features include:

• Separate Numeric Keypad simplifies data entry
• Status Lights for Caps, Num and Scroll Lock helps eliminate typing errors
• Extra-wide Control and Return Keys for ease of use
• Return Key placed in "standard typewriter" position for touch typists

BORLAND'S AWARD-WINNING SUPERKEY INCLUDED! GET THIS BEST-SELLING KEYBOARD ENHANCER SOFTWARE — VALUED AT $69.95 — AS PART OF THIS AMAZING OFFER AND CAPTURE THE POWER OF MACROS!

SuperKey's macro capability is remarkable. It literally turns a thousand keystrokes into one.

• Exclusive screen data Cut & Paste
• Screen Privacy and Protection
• Time & Data Stamping
• ProKey Compatibility
• Data Encryption

ORDER NOW — CALL TOLL FREE 800-826-5398; IN CA 800-592-9602

PLEASE RUSH ME KEYBOARD/SUPERKEY BUNDLES AT $99.95 EACH. PLUS $10 SHIPPING AND HANDLING IN THE U.S.A. *

ENCLOSED IS MY CHECK FOR $109.99 FOR EACH BUNDLE ($116.45 FOR CA RES. — SALES TAX INCL.)

EXP. DATE SIGNATURE

NAME

ADDRESS

CITY STATE ZIP

TELEPHONE COMPUTER TYPE & MODEL

* PC Jr., AT&T and Corona computers require adaptors available at a nominal fee. Call for pricing.

CRITICS' CHOICE

"Who Can Pass Up a Deal Dept?"
John C. Dvorak, InfoWorld, 3-10-86

"The best part of the keyboard is the feel. It is ideal!"
PC Productivity Digest, 5-86

"The keyboard's construction is solid...also quick and responsive to the touch. It is ideal for anyone who wants to or has to replace their PC keyboard...At $129.95 DataDesk's keyboard is a bargain. The PC 8700 keyboard with SuperKey for that price is a steal."
Computer Shop Talk, 3-3-86

"DataDesk International has designed a sturdy and handsome keyboard that has a tactile response...and bundled with SuperKey is the PC hardware bargain of the year."
Charles Humble—Computer Editor, The Oregonian

"DataDesk's is genuinely a nice keyboard...It feels nice...I can type much faster and much more accurately on it than most others...and it's inexpensive. It costs only $129, compared to around $300 for the Keyboard 5151."
Harry Newton, Publisher, Teleconnect, 5-86

RISK FREE 30-DAY MONEY-BACK GUARANTEE

NOW FOR YOUR IBM PC, XT, AT, AT&T, PC Jr. AND COMPATIBLES *