

CIRCUIT CELLAR **I N K**®

THE COMPUTER
APPLICATIONS
JOURNAL

**BONUS
SECTION**

**Embedded
Applications**

Graphics

Creating
Fractal Worlds

The Magic of
Biomorphs

Build a
V25-based
Controller

October/November 1990 — Issue 17

\$3.95



CASE is Coming

EDITOR'S INK

Curtis Franklin, Jr.

It's fun to watch events come together to form a trend. A while back, I told you that events were coming together that would result in a trend toward using "PC-compatible" platforms for control applications in ever-increasing numbers. Things are still rolling along with that trend, and I feel pretty good about the prediction. I feel so good, in fact, that I'm going to hit you with another prediction this time: I predict that most of the people reading this will be using **CASE—Computer-Aided Software Engineering**—within the next **five years**. It doesn't matter whether most of your programming time is spent on desktop computer applications or embedded control, I believe you will start using CASE tools. I'm making this prediction based, not on any sort of blazing insight, but on the logical progression of several industry trends.

The first important trend is pretty obvious: Programmers are moving away from assembly language and toward **high-level** languages for programming. No one who does most of their programming for personal computers will be surprised at this, but assembly programming is still SOP for much embedded and control application software. The trend really started quite a few years ago, when BASIC and FORTH were put into on-processor ROM for the **Z8**. They weren't, of course, "real" programming languages, and most people only used them for prototyping, but their relative ease-of-use was seductive. Before long, engineers were looking for excuses to use BASIC rather than assembly language, and trying to extend the higher-level languages for greater flexibility. A bit of time passed, and separate **high-level**-language compilers and interpreters began to appear for traditional controller chips. These were all cross-compilers, most with limited libraries, and almost all had user interfaces that were abysmal by any standards but their own. Nonetheless, they worked, after a fashion, and they helped to separate the programmer from time-consuming tedium. The difficulty and frustration encountered in using these tools were seen by many engineers as a sort of "ennobling pain," a badge of honor for those intellectually macho enough to tolerate them. In the mean time, programming tools for personal computers were developing along somewhat different lines.

Personal computers tended to have increasing levels of processor power and memory capacity. These were used by compiler, debugger, and editor writers to support greater numbers of functions and higher levels of integration. Rather than viewing struggle with tools as an honorable pursuit, desktop application programmers saw primitive tools as impediments to greater productivity and better software. They were particularly interested in seeing software development tools keep pace with the rising levels of power and functionality available from their hardware. Compiler and development tool vendors responded,

and the programmer's "quality of life" improved. Finally, there came the trend that would tie everything together.

Intel pushed to merge desktop microcomputer and embedded **controller architectures with the 80186**. Here was a microprocessor that was code-compatible with the popular **8088** and **8086** microprocessors, yet had I/O features more typical of embedded controllers. Through time, the price of IBM PC/XT-clone motherboards dropped to a point where the same I/O, operating system, and memory architectures could be economically used for both desktop computing and control applications. Engineers and control programmers have been introduced to the wonders of the modern desktop development environment, and the steps are short between (for example) the Borland Turbo-language environment and CASE.

In CASE, the programmer writes no code, but a detailed specification of the program required. All of the unpleasant (and time-consuming) details of interrupt servicing, I/O generation, and housekeeping are handled by the CASE software. While the most obvious current applications of CASE are in GUI-software development (where the programmer does not want to spend endless time re-creating the graphical shell for each program), the tools are rapidly progressing to the point where they will be coveted by most programmers. Why? The basic reasons are that they save programmer time and take some of the repetitive tedium out of programming.

It's no longer news that engineering and programming time is more valuable than hardware. The trouble is that the gap is becoming so wide that employers and clients will push for anything that cuts down on time required for a project. CASE fits that bill nicely. Now, the support for CASE is not universal. On our engineering staff, Ed Nisley tells me that it takes as much time to write a detailed specification as to write the code. He's right, of course, but it just doesn't matter. CASE will come to your software environment, and I'm betting that you'll be glad when it does. The force of history is on my side, and (ultimately) we'll see better and more creative applications because of it.



FOUNDER/
EDITORIAL DIRECTOR
Steve Ciarcia

PUBLISHER
Daniel Rodrigues

EDITOR-in-CHIEF
Curtis Franklin, Jr.

PUBLISHING
CONSULTANT
John Hayes

ENGINEERING STAFF
Ken Davidson
Jeff Bachiochi
Edward Nisle y

CONTRIBUTING
EDITOR
Thomas Cantrell

NEW PRODUCTS
EDITOR
Harv Weiner

CONSULTING
EDITORS
Mark Dahmke
Larry Loeb

CIRCULATION
COORDINATOR
Rose Mansella

CIRCULATION
CONSULTANT
Gregory Spitzfaden

ART & PRODUCTION
DIRECTOR
Tricia Dzedzinski

PRODUCTION
ARTIST/ILLUSTRATOR
Lisa Ferry

BUSINESS
MANAGER
Jeanne Walters

ADVERTISING
COORDINATOR
Dan Gorsky

STAFF RESEARCHERS

Northeast
Eric Albert
William Curlew
Richard Sawyer
Robert Stek

Midwest
Jon Elson
Tim McDonough

West Coast
Fran Kuechmann
Mark Voorhees

Cover Illustration
by Robert Tinney

CIRCUIT CELLAR **INK**[®]

THE COMPUTER APPLICATIONS JOURNAL

In This Issue...

EMBEDDED APPLICATIONS SECTION

43

Multichannel Digital Voltmeter Interface

MAX134 Chip Adds High-Performance ADC to Embedded Control

by Steve **Ciarcia**

An autoranging DVM interface is the first step in Ciarcia's newest building automation plans.

58

Control Theory for Embedded Controllers

An Introduction to the Basics of Computerized Control

by Thomas **Mosteller**

Knowing how a system should work is the key to building working embedded systems.

67

Using C For Embedded Control

*Building a 6805-Based
Darkroom Timer*

by **Ashok Patel & Walter Banks**

A high-level language can make your development time more productive.



FEATURES

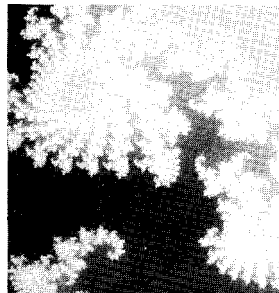
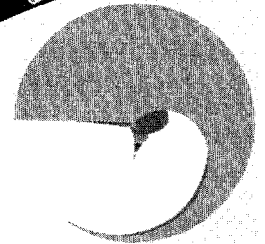
16

Functions of Complex Variables

Generating Biomorphs on Personal Computers

by **Saim Ural**

Fractal-like methods can create beautiful "biomorph" patterns and images.



24

Creating Fractal Images

Using the Power of Fractals for Realistic Planetary Images

by **Chris Ciarcia**

Extend fractal techniques to model realistic planets and landscapes.

38

Running VGA on an IBM Professional Graphics Display

by **J. Conrad Hubert**

You can connect affordable workstation monitors to VGA adapters with a simple hardware addition.

100

The Second Circuit Cellar INK Design Contest Winners

DEPARTMENTS

1

Editor's INK
CASE is Coming
by **Curtis Franklin, Jr.**

5

Reader's INK-Letters to the Editor

10

NEWProductNews

73

Firmware Furnace
The Furnace Firmware Project, Part 3
light Code Meets the C Monster
by **Ed Nisley**
The Furnace Firmware project continues with additions to the acquisition and display hardware.

82

From the Bench
PC Programming Comes to Embedded Control
V25...An 8088 with all the good stuff
by **Jeff Bachiochi**
A new processor board! Jeff shows how to build an 8088-compatible (V25-based) controller.

89

Silicon Update
VHDL-The End Of Hardware?
by **Tom Cantrell**
Are we looking at the end of hardware? New chips put circuit design as we know it on the endangered list.

98

Practical Algorithms
Around and Around We Go...
by **Scoff Robert Ladd**
Is recursion or iteration the way to go when you need to QuickSort your data?

104

ConnecTime—Excerpts from the Circuit Cellar BBS
Conducted by **Ken Davidson**

112

Steve's Own INK
A Computer is A Computer
by **Steve Clarica**

97

Advertiser's Index

Circuit Cellar BBS-24 Hrs.
300/1200/2400 bps, 8 bits, no parity, 1 stop bit, (203) 871-1988.

The schematics provided in Circuit Cellar INK are drawn using Schema from Omaton Inc. All programs and schematics in Circuit Cellar INK have been carefully reviewed to ensure that their performance is in accordance with the specifications described, and programs are posted on the Circuit Cellar BBS for electronic transfer by subscribers.

Circuit Cellar INK makes no warranty and assumes no responsibility or liability of any kind for errors in these programs or schematics or for the consequences of any such errors. Furthermore, because of the possible variation in the quality and condition of materials and workmanship of reader-assembled projects, Circuit Cellar INK disclaims any responsibility for the safe and proper function of reader-assembled projects based upon or from plans, descriptions, or information published in Circuit Cellar INK.

CIRCUIT CELLAR INK (ISSN 08968985) is published bi-monthly by Circuit Cellar Incorporated, 4 Park Street, Suite 20, Vernon, CT 06066 (203) 875-2751. Second-class postage paid at Vernon, CT and additional offices. One-year (6 issues) subscription rate U.S.A. and possessions \$14.95, Canada/Mexico \$17.95, all other countries \$26.95 (surface). \$38.95 (air). All subscription orders payable in U.S. funds only, via international postal money order or check drawn on U.S. bank. Direct subscription orders to Circuit Cellar INK, Subscriptions, P.O. Box 3050-c, Southeastern, PA 19398 or call (215) 630-1914.

POSTMASTER: Please send address changes to Circuit Cellar INK, Circulation Dept., P.O. Box 3050-C, Southeastern, PA 19398.

Entire contents copyright ©1990 by Circuit Cellar Incorporated. All rights reserved. Reproduction of this publication in whole or in part without written consent from Circuit Cellar Inc. is prohibited.

Letters to the Editors

PRACTICAL ALGORITHMS

I've been a subscriber since I found out about CIRCUIT CELLAR INK and love the magazine, but I'm truly concerned about Scott Robert Ladd's "Practical Algorithms" column. I know that solder is Steve's favorite programming language, but for many reasons Modula-2 just isn't an appropriate language for the CIRCUIT CELLAR INK audience.

I strongly urge Scott Ladd to switch to C from Modula-2 for the following reasons:

- C is available on more (many) processors.
- CIRCUIT CELLAR INK is aimed at people who need to bit-twiddle and are not afraid of bit-twiddling. C is perfect for bit-twiddling-Modula-2 is not.
- While it's true that C is not the best language for teaching everything to everybody, it should be a good language for teaching CIRCUIT CELLAR INK's target audience.

*Although I've never taught Modula-2, I taught Pascal for three semesters in a local university. I've used Pascal to write a cross-assembler for the 6502. These two things taught me that I'd never Modula-2) Pascal (or C were also available.

*An example: I recently saw an article about ANDs and ORs in Modula-2 with the impression that turning our software weird stuff you have to do in order to AND two & ORs is not the least-significant bit in variable *i*. What could be simpler?

Thanks for a great magazine and for your consideration of this topic.

[Editor's Note: This letter is taken from parts of separate letters sent to CIRCUIT CELLAR INK and Scott Ladd.]

Don Lasley
Cordova, TN

Scoff has a brief reply to this concern in his article on page 98. Scoff and I discussed the language for his column before the first article was written and decided against using C for one reason: The purpose of the column and C don't match.

Scoff's column is "Practical Algorithms." That means he writes about the logic of a solution, not the implementation of that solution. This requires a language that is clear, simple, and readily translated into the languages of implementation. Modula-2 is, in our opinion, the best fit for this purpose, followed by Algol and Pascal. C is not even in the top five contenders as a language for expressing an algorithm.

The same qualities that make C a good choice as a programming language (terse syntax and rich semantics) make it an exceptionally poor choice for expressing algorithms. If enough readers ask for a column on programming in C, we'll start one, but "Practical Algorithms" will continue to use the best fool for

Curtis Franklin, Jr.

MORE ON REMOTE CONTROL

I was both disturbed and mystified by the opening remarks in the article "ONDI-The ON-line Device Interface" (CIRCUIT CELLAR INK #16). John Dybowski's statement that the requirements of software like our Remote-Console system can "cause serious problems" leaves the impression that turning our software is something they should avoid. To the best of my knowledge, aside from the added wear on bearings in the hard disk drive when a computer is left running while unattended, there is no problem. In fact, not turning the computer off should actually be beneficial to the other components as this avoids the problem of thermal shock-one of the major causes of component failure in electronic equipment.

Beyond the opening paragraph, I did find the article interesting. The device solves two of the problems I do have. Those are the problems of rebooting a computer from a remote location and turning it off when you no longer wish to keep it on-line. Those are the two things that can't be done through software. The Remote-Control system has a SHUTDOWN command that removes it from memory, but leaves the system running. Rebooting PCs

remotely through software isn't possible because of a flaw in most BIOSs, even though there is a DOS function (INT19) that is supposed to accomplish this—it doesn't work on most machines.

On the other hand, the Remote-Control software solves a problem the ONDI hardware can't even address. That is, the problem of duplicating the host console screen on the remote computer's screen which is what my software is really all about. The CTTY command doesn't really work. It should, but most application software is so ill-behaved that it can't work. I know, because trying to use the CTTY command (with an H-19 terminal) is what prompted me to write the first version of Remote-Console.

I hope that you will publish a clarification of the opening paragraph of this article, explaining to your readers that running the Remote-Console software poses no real danger to their equipment.

Louis E. Wheeler
Oceano, CA

John Dybowski replies:

In the opening statement of my article on computer remote control I stated, in reference to remote control software in general, "All require the host computer to be powered up and running special software to be accessible. These requirements can cause serious problems when remote control is required..." I hope it is evident that I was alluding to the inconvenience of having to leave the computer powered up and running special

software, and the danger of the computer becoming inaccessible due to phenomena such as power line glitches or dips. (For example, most PCs will not reboot properly following a short power outage.)

I do not believe that the use of remote control software products such as Remote-Console can pose any danger to computer equipment. Finally, it was never my intention to offend, or to understate the usefulness of remote control software. In fact, my opinion on this subject is quite the contrary. I feel that the most powerful remote control capabilities can be attained by combining hardware and software. Due to the inherent strengths and limitations of the respective disciplines, hardware and software-based remote control techniques do not compete, they complement one another.

CHIP EQUALITY!

I would like to reinforce the comments of Chuck Yerkes' ("Reader's INK," CIRCUIT CELLAR INK #11) and Michael Black's ("Reader's INK," CIRCUIT CELLAR INK #15) letters. CIRCUIT CELLAR INK seems to be ignoring the 65xx and 68xx family of processors.

I would very much like to see articles based on the 6502 and 65C816 microprocessors. Would CIRCUIT CELLAR INK please make a special effort to collect and publish articles on these chips?

Also, is there any chance of CIRCUIT CELLAR INK attracting advertisements from suppliers that would offer computer designs based on the 6502 or 65C816 family similar to what the magazine now has for the 8031 and 8051 processors? Are there any suppliers reading this letter who would like to sell me a processor card based on the 65C816? How about a cross-assembler for the PC with a 65C816 as the target?

Joseph Ennis
Valparaiso, FL

We know it may seem hard to believe, but there is truly no architecture bias in the editorial office. There is, however, a strong bias toward printing articles based on working hardware; so far, most of those articles have centered on Intel microprocessors. The engineering staff has recently completed the design for a board based on the 68HC11 microprocessor—we hope to print an article based on that design in the near future.

Dan Rodrigues, CIRCUIT CELLAR INK's publisher, assures me that he is actively seeking advertisers for products across all microprocessor architectures. He suggests that recent advertisers Avocet, Micro Dialects, PseudoCorp, or Universal Cross Assemblers might well have the cross-assembler you are looking for.

Editors



TAKE A LOAD OFF...

A rugged CABBAGE CASE? lined with plenty of foam for your equipment can **TAKE A LOAD OFF YOUR MIND** when you've got to travel.

TAKE A LOAD OFF YOUR BACK with our exclusive tilt-wheels and extension handle option.

UNLOAD ON US!
Call or write to tell us about your shipping or carrying problems
WE HAVE SOLUTIONS!

CABBAGE CASES
CABBAGE CASES, INC.
1166-C STEELWOOD ROAD
COLUMBUS, OHIO 43212-1356
(614) 466-2495 FAX (614) 466-2788
(800) 888-2495

Reader Service #116

A-BUS™ NEWS

New Products

Alpha Products proudly announces two new product lines: **C-Net** serial communications devices, and Alpha **Box** interfaces. These new products are not merely **A-Bus** accessories, but complete sets of products for all of your interfacing needs.

All the products are used to connect different types of devices to your computer. Our communications devices help you connect devices that have computer interfaces already built in. **C-Net** provides the option of connecting many different RS-232 devices to a single serial port on your computer. We also carry converters to other standards, including RS-422, RS-485 and IEEE-488.

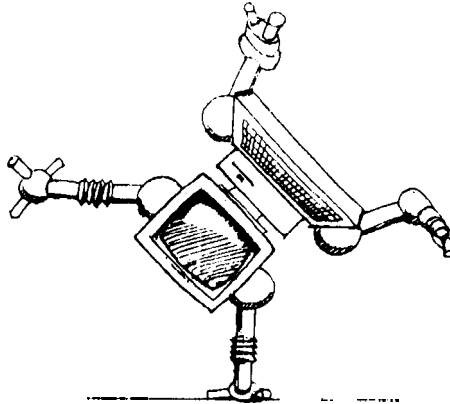
- **C-Net Adapter**. Connects the master control computer to **C-Net**. \$74
- **Quad C-Net Module**: Connect 4 RS-232 serial devices to **C-Net**. Each device is configurable (baud rate, parity, etc.) and has **4.8K** byte input and output buffers. \$695
- **C-Net Device Module**: Connect any RS-232 Device to **C-Net** for data collection or communication, with handshaking. \$195

Alpha Boxes and **A-Bus** cards both provide ways to interface other types of devices to your computer. Alpha **Boxes** sense, measure, switch and govern. They feature:

- Each box is an attractively packaged self contained module that connects directly to the computer and includes power supply.
- The input boxes offer the option of logging data "off-line" and downloading it rapidly to the computer.
- Built-in intelligence provides a simple and consistent interface to your software.

A Sampling of Alpha Box Products:

- **Digital Input**: 64 **TTL/CMOS/0.5V** input channels. 6495
- **Digital Output**: 64 **TTL/CMOS/0.5V** level outputs. 6495. **120VAC** control available.
- **Digital I/O**: 32 **TTL Level (0.5V)** Inputs and 32 outputs. 6495
- **Analog Input**: 16 channels. **0-5.1V, 20mV** steps (8 bit). **2000 readings/sec.** 6495.
- **Expansion Option**: 16 more channels. \$100
- **12 Bit Analog Input**: 16 channels, programmable gain. **10000 inputs/sec.** max. \$995. Option: 16 more inputs. \$200
- **Analog Output**: 4 channel, 12 bit D/A. **±5.1V** outputs. 6495. Expander Option: 12 more outputs. \$296
- **Counter**: 16 inputs, 24 bit. 6595



"We can make your PC do things you wouldn't believe."

C³ From Your PC

- **Command**
- **Control**
- **Communications**

Bring new dimensions to your computer with **A-Bus, C-Net** and Alpha Boxes. No longer is your computer limited to number crunching or word processing. Now you can connect it to all kinds of equipment, sensors or machines. This offers unprecedented power from production lines to experiments to home control.

Each product is designed to fit your needs: They're affordable. Compare our prices: the cost of a solution is surprisingly low. They're simple and easy to connect to your computer and your application, and carefully designed to adapt to your software easily. They're **versatile**. An infinite number of combinations is possible: one of them is right for you. Easily expanded or changed for future projects. They're proven. We have customers around the world, including Fortune 100 companies, most major universities, governments and individuals.

Call for a Catalog (800) 221-0916

Overseas distributors

UK: Pinna Electronics, Scotland
Tel: (6294) 695296 Fax: (9294) 68286
Asia: **Batam** DA, Singapore
Tel: 473-4518 Fax: 479-6496
Scandinavia: **A/S** Con-Trade Norway
Tel: (04) 41 83 51 Fax: (04) 41 94 72

Low cost Data Acquisition and Control

A-Bus Sensing & Measuring:

- Read switch status. Detect or measure voltage. Read pressure, temperature, weight and other sensors. For example:
- High-Speed **12-bit** A/D converter: 8 **10µs** analog inputs. 1 **mV** resolution \$179
 - 8 Bit A/D: 8 inputs, 0-5.1V in 20mV steps, 7500 **conversions/sec.** \$142
 - 12 **Bit** A/D: **±4V** in 1 **mV** steps, 130mS conversion time. 1 input, expandable \$153
 - Temperature Sensor: **0-200°F** 1° Accuracy. **10mV/°F.** \$12
 - Digital Input: 8 opto-isolated. Read voltage presenceswitch closure. \$65
 - Latched Input: Each individually latched to catch switch closures or alarm loops. \$85
 - Touch Tone Decoder: \$87
 - Counter/Timer: 3 **16-bit** counters. Generate or count pulses. Time events. \$132
 - Clock with Alarm: real time clock with calendar and battery backup. \$98

A-Bus Switching & Governing:

- Switch any type of electrical device. Adjust level or position. A sampling:
- Relay Card: 8 individually controlled industrial relays. 3A at **120VAC**, SPST. \$142
 - Digital Output Driver: 8 outputs: 250mA at 12V. For relays, solenoids... \$78
 - Reed Relay Card: 8 individually controlled relays. **20mA @ 60VDC**, SPST. \$109
 - Multiplexer: Switch up to 32 channels to a single common. \$63
 - Smart Stepper Motor Control: Microprocessor controls 4 motors. English commands for position, speed, units, limits, etc. \$299
 - Telephone Control Card: On/off hook, generate and decode touch tones, call progress detection. \$159
 - X-10 Controller: Control and sense standard wall outlet power modules. \$149
 - Voice Synthesizer: Unlimited vocabulary, text to speech software built in. \$159
 - D/A: Four 8 Bit Outputs. Adjustable full scale. \$149
 - 24 line **TTL** I/O: Connect 24 signal. **TTL 0/5V** levels or switches. (8255A) \$72

A-Bus Adapters and Software:

- Adapters connect **A-Bus** cards to your particular computer.
- Plug-in adapters for IBM **PC/XT/AT/386** and compatibles (\$69). Micro-Channel (\$93). Apple II. Commodore, TRS-80.
 - Serial adapters for Mac, PC, etc.
 - Odin PC compatible software. Control relays from analog inputs or time schedules. Logging. Runs in background. \$129

ALPHA Products

242-C West Ave, Darien, CT 06620 USA (203) 656-1 806 Fax 203 656 0756

Reader Service #105

October/November 1990 7

COMPLEX VIDEO

I have a project at work which requires a PC to display two different sets of textual information on two different monitors at the same time. I realize this is a simple task if I use a monochrome video card for one display and a CGA card for the other. The catch here is that I need two composite outputs which will control up to eight displays each. I do not believe I can put two CGA cards in one machine, and I have never seen a mono card with a composite output. As I see it now, I need one of four things:

1. A mono card with a composite output;
2. A TTL mono monitor with a composite "tap" connected to a normal mono card;
3. Some way to accomplish #2 with additional circuitry;
4. A converter to change the TTL output of the mono card into composite video.

Additionally, since I am driving eight monitors over a distance of 400-500 feet (total) I need some kind of line amplifier to boost the composite signals. Each display must be capable of 80 x 25 text; no graphics are required.

This project is at the top of my priority list. If you could provide some answers I would greatly appreciate it.

Dick Dasinger
Bismark, ND

This sounds like an interesting project! Let's see if we can point you in the right general direction..

First of all, give Black Box Corporation a call (412/746-5565) and talk to them about their video splitters for RGB and composite signals. From what I saw in their catalog, they don't have what you want but they may, as the saying goes, have what you need.

As far as getting dual video out of your system, I'd make a teeny modification to a pair of CGA cards and a little change to your code. First, recall that the CGA video buffer starts at B8000 and uses I/O ports 3D0-3DF. When you use two cards, you must make sure they use different video and I/O addresses.

The BIOS stores the following information in RAM in segment 0040:

- 49—display mode (byte)
- 4A—number of columns
- 4C—length of video buffer in bytes
- 4E—video buffer address (segment)
- 50-5F—cursor position for all eight pages
- 60—cursor type
- 62—current display page (byte)
- 63—video controller I/O address
- 65—current 3x8 register setting (byte)
- 66—current 3x9 register setting (byte)
- 84—number of video rows - 1 (byte)
- 85—character height in bytes/character
- 87—video control states 1 (byte)
- 88—video control states 2 (byte)

Now, here's the deal. If you modify a card to put it at differ-

Science, Engineering & Graphics Tools for MS C, MS Quick C, MS Fortran, MS QuickBasic, Turbo C, Turbo Pascal

The Science/Engineering/Graphics Tools (Revision 7.0) are a collection of general purpose routines which solve the most common data analysis and graphics problems encountered in science and engineering applications. *All* of the routines are supplied on disk in the source code of the target language and can be used royalty free when compiled into an application program. A 200 page manual describes each function in detail.

These tools are available for Turbo Pascal, Turbo C, Microsoft C and Quick C, QuickBasic and Microsoft Fortran for IBM compatibles.

Ordering Information

Model#	Version	Price
IPC-TP-016	Turbo Pascal 4.0, 5.x	\$100
IPC-TC-006	Turbo C V 2.x, Ctt	\$100
IPC-MC-006	Microsoft 5.x, 6.0 & Quick C	\$100
IPC-QB-006	QuickBasic V 4.x	\$100
IPC-MF-006	Microsoft Fortran V 5.0	\$150

Shipping charge is \$4 within USA, \$7 Canada. Elsewhere add \$22 for shipping. Mastercard, Visa, Company PO's, and personal checks accepted. MASS residents add 5% sales tax. Credit card orders please include expiration date of card.

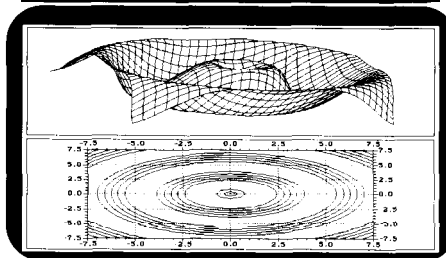
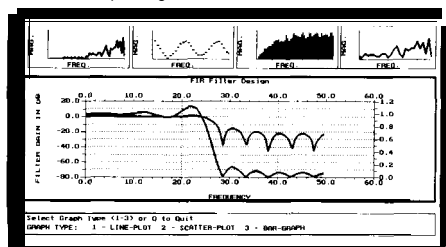
FEATURES

- 100% Royalty Free
- 100% Source Code

CRT Graphics Adapter Support - the graphics libraries use the graphics routines supplied with the respective compiler. (CGA, EGA, Hercules, VGA)

Now includes PostScript and HPGL File support

Science/Engineering charting routines Linear, semi-log, and log graphs. Auto-scaling of axes, line, scatter, error bar, pie, bar charts and contour plotting. Multiple x and y axes for the same graph. Hardcopy support PostScript and HPGL support direct to a printer/plotter or to a file. Raster screen dumps to the Proprinter, Epson MX, FX and LQ printers, HP Laserjet and Thinkjet printers. 3-D plotting translation, scaling, rotation, and function plotting routines



Statistics - mean, mode, standard deviation, etc.
Multiple Regression - With summary statistics
Curve Fitting - Polynomial and cubic splines
Simultaneous Equations - real and complex
Fourier Analysis Forward and inverse FFT,
Windows, 2-Dimensional FFT, Power Spectrum,
FIR Digital Filtering

Matrix & Complex Number Math
Eigen values and vectors - Cyclic Jacobi
Integration and Data Smoothing
Differential Equation - Runge-Kutta-Fehlberg
Root Solving - Bisection, Newton and Brent roots
Special Functions - Gamma, Beta, Bessel, etc.
RS-232 Support all versions include an interrupt driven RS-232 driver

Numeric Types C versions now compatible with float and double numeric types. Pascal version compatible with Real, Single and Double types.

Ask about our new Microsoft C and Turbo C Tools package for the Metrabyte Das-8 and Das-16 data acquisition and control boards



21 Highland Circle, Needham, MA 02194 USA
Tel. (617)449-6155 FAX (617)449-6109

ent RAM and I/O addresses, I think all you need to do is set the information at the locations shown above and the standard BIOS will talk to the card! Because you're just updating a few RAM locations, swapping monitors is quick and easy, yet because you're using the standard BIOS to talk to the cards, everything will continue to work just fine.

Regardless of what kind of CGA card you've got, there will be an address decoder on it somewhere that looks at the incoming address lines and decides when the card is selected. What you want to do is change the address decoder to move the card from B800:0000 to B000:0000 and switch the I/O addresses from 3D0 to 3C0. The RAM address is normally the monovideo buffer

and the I/O address is normally an EGA card, but what the heck, they're sure to be unused in your system.

You might want to pick another segment for the video buffer so you can continue to use the monochrome card in the system. Imagine that... three monitors from one box!

If your CGA card uses discrete logic gates, you can flip A15 at the video buffer and A4 at the I/O decoder; you will probably have to glue an inverter chip on the back of the card. If your card uses a fat LSZ chip with the decoders inside, buy another card with older and simpler logic. In any event, make sure you don't scramble the address lines used elsewhere on the card. Take note of the traces that run under the chips, because A4 (for example) is used to address the video buffer RAM, too.

Depending on how you've structured your code, the required changes are either trivial or agonizing. Because the BIOS will initialize only the standard CGA card, you'll have to do the setup for the other one. You also need to call a routine to swap the RAM data whenever you change monitors: If you have a centralized routine to update the displays this can be a single line of code.

We can't put everything into this letter, so here's a required reading list:

"The PC Programmer's Sourcebook"

Thorn Hogan

Microsoft Press

ZSBN 1-55615-118-7

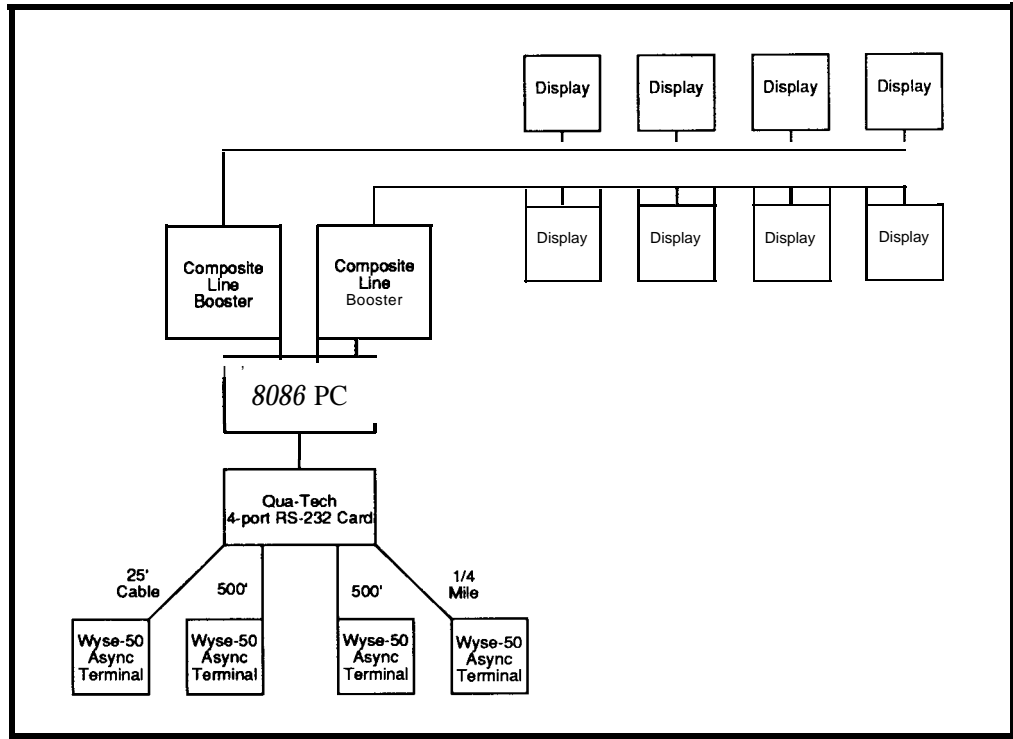
Summarizes all the ports, pins, addresses, and such like that you'll find in a dozen other references. No text, just an awesome collection of tables.

"Programmer's Guide to PC & PS/2 Video Systems"

Richard Wilton

Microsoft Press

ZSBN 1-55615-103-9



Gives the straight dope on all the video adapters you'll ever need to use. Good programming examples, good writing.

Good luck, and tell us how it works out!

OrCad Users

Discover a **menu** system that **saves you time and increases productivity.**

The **Intelligent Menu System** ties together all utilities with a user-friendly, pop up menu. And **IMS** does not impact working RAM space.

Start using powerful new features:

- Plot spooler
- File/view editor
- Directory manager
- On-line help system
- Easy custom configuration
- Stuff file builder

IMS

The 'Intelligent Menu System'

Call now for a **FREE** demo disk

1-800-966-8856



31% Unit A, East La Palma, Anaheim, CA 92806

We guarantee satisfaction or your money back

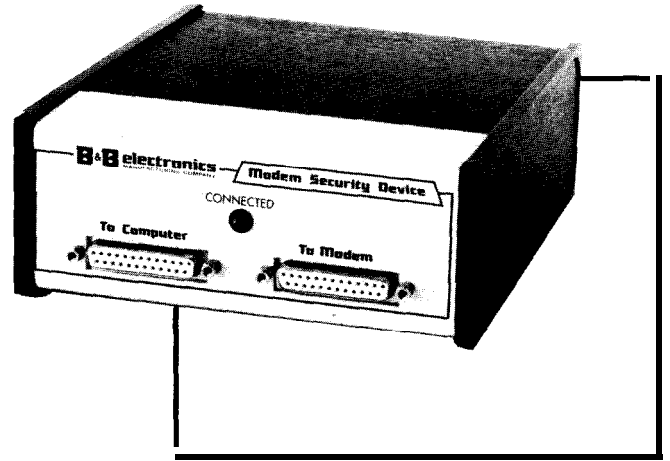
MODEM SECURITY DEVICE

Protecting a computer or private bulletin board from unauthorized entry or computer viruses can be accomplished with a Modem Security Device (MSD) from B&B Electronics. The Model 232MSD uses the call-back method to assure proper entry. A caller must not only have a correct password, but must be located at the correct phone number.

The 232MSD works with most stand-alone modems that are "AT Command Set" compatible. When a caller reaches the modem, the MSD intercepts the call and asks for the password. Upon receipt of a valid password, the MSD will wait for the caller to hang-up, and call the

phone number stored in its memory that corresponds to that password. The user at that number is then allowed access to the computer. If someone steals a user password, it can't be used because they will not be at the user's phone number.

The 232MSD features a 50-number directory of password/phone number combinations that is stored in EEPROM to avoid loss during a power failure. A built-in password/phone number editor can be accessed locally, and the System Manager can give a user password access without the call-back requirement. Calls originating at the computer are not blocked. The unit also features automatic data rate selection of rates from 300 to 19,200.



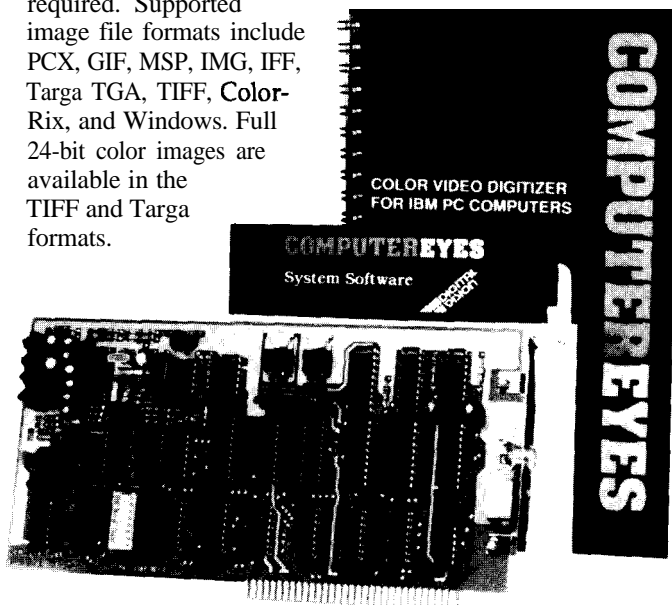
The Model 232MSD sells for \$149.95.

B & B Electronics
4000P Baker Road
P.O. Box 1040
Ottawa, IL 61350
(815) 434-0846
Fax: (815) 434-7094

Reader Service #214

HIGH-RESOLUTION VIDEO DIGITIZER

The ability to capture high-resolution images (640 x 480) from any standard (or Super VHS or Hi-8) camcorder or VCR at up to 24-bit (16 million color) palette depth is offered by the **ComputerEyes/Pro** from Digital Vision Inc. Captured images are displayed with the PC's standard EGA, VGA, or Super VGA graphics capabilities, with no additional hardware required. Supported image file formats include PCX, GIF, MSP, IMG, IFF, Targa TGA, TIFF, Color-Rix, and Windows. Full 24-bit color images are available in the TIFF and Targa formats.

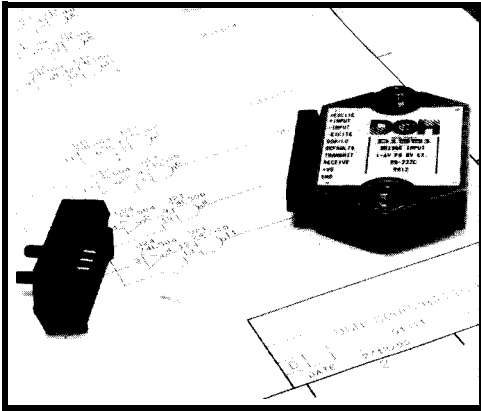


The 8-inch-long board requires only one 8-bit slot on the PC bus and accepts either standard NTSC composite color (or b/w) video or S-Video (separate luma and chroma). It features simultaneous capture and display with image scan times from 1.5 to 24 seconds. A simple, fast, live-image "preview" mode to frame, focus, and adjust color and intensity balance before capturing is provided. Also included is an advanced palette selection routine to optimize colors for 256-color or 16-color display modes. Image enhancement features include color palette adjustments, image sharpen and smooth routines, image scroll, shrink and expand.

The **ComputerEyes/Pro** requires an IBM PC, XT, AT, 386, 486, or compatible or PS/2 Model 25/30 with 640K RAM minimum; MCGA, EGA, VGA, or Super VGA graphics; one 5.25" floppy drive; and DOS 2.1 or higher. The unit sells for \$449.95.

Digital Vision, Inc.
270 Bridge Street
Dedham, MA 02026
(617) 329-5400

Reader Service #215



PRESSURE SENSORS

Pressure measurements with a computer have been simplified with analog-to-RS-232/RS-485 modules announced by DGH Corporation. The **D1550/D1560** and **D2550/D2560**

make it easy to interface 1-6-VDC pressure sensors to any computer with a serial port. The DGH modules combine wide-range analog input signal conditioning, sensor excitation, A/D conversion, on-off control features, and communications in ASCII over an RS-232 or RS-485 link. The **D2550/D2560** are enhanced versions of the **D1550/D1560** that provide the ability to scale the output to desired engineering units. The series has eight models including 1-6-V

bridge input with 8-V excitation, and 1-6-V bridge input with 10-V excitation, with either RS-232 or R'3-485 output. Input burnout protection is 250 VAC. Measurement resolution is one part in 50,000 and the conversion rate is eight conversions per second. Accuracy is $\pm 0.05\%$ of full scale maximum.

The modules use an 8-bit CMOS microcomputer to perform all scaling, linearization, and calibration tasks in software, eliminating the need for pots, switches, or adjustment hardware.

The singlechannel data acquisition modules eliminate multiplexing problems by putting the hardware at the input source. In a typical distributed application, pres-

ures are measured in several remote sites while the data is monitored and controlled from a more convenient location. Up to 124 modules can be strung on a single set of wires.

Communication features include channel address, data rates from 300 to 38.4k bps, parity, line feed, byte time delays, echo, and checksum. The modules store communications setups in nonvolatile memory.

The **D1550/D1560** modules sell for \$325.00 and the **D2550/D2560** modules sell for \$350.00.

DGH Corporation
P.O. Box 5638
Manchester, NH 03108
(603) 622-0452
Fax: (603) 622-0487
Reader Service #216

RAMIFIED REAL-TIME CLOCK CHIP

A real-time clock with 4096 bytes of nonvolatile memory has been developed by Dallas Semiconductor. The **DS1387 RAMified Real-Time Clock** is compatible with PC hardware and software operating systems. It merges the 4K bytes of SRAM into the PC without hardware or software compatibility



problems for the AT, PS/2, and EISA buses.

The **DS1387** is a self-contained subsystem that includes a lithium power source, quartz crystal, and CMOS chip. It counts seconds,

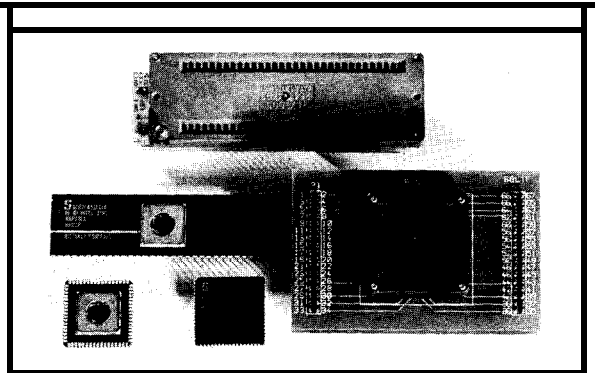
minutes, hours, day of the week, date, month, and year with leap year compensation. A unique feature of the device is a "freshness seal" that prevents lithium power consumption until the PC is first powered up. Thereafter, no lithium power is consumed as long as system power is present. This feature eliminates unnecessary drain on the energy source during shipping and storage, as well as during PC use, and allows a useful life of more than 10 years.

The additional memory can be used to store machine-specific information such as serial number, password, component status, type of add-in boards, and so on.

The **DS1387 RAMified Real-Time Clock** sells for \$13.75 in 1000-piece quantities.

Dallas Semiconductor
4350 Beltwood Parkway S.
Dallas, TX 75244-3219
(214) 450-0400 • Fax: (214) 450-0470

Reader Service #217



87C451 PROGRAMMING ADAPTERS

As embedded control applications become more complicated, designers are moving to bigger and better microcontrollers. The **87C451** is an expanded I/O version of the **87C51** and is available in both a 68-pin PLCC and 64-pin DIP version. Since most EPROM programmers only support up to 40-pin devices, programming the **87C451** has been difficult.

The **PA451-68** and **PA451-64** programming adapters from Logical Systems Corporation provide a low-cost means of program-

ming these devices. The **PA451-68** accepts a PLCC or ceramic LCC **87C451** microcontroller and plugs into a 40-pin programmer as if it were an **87C51**. A push-in/pop-out socket was selected for positive alignment and ease of use. All contacts are gold. The **PA451-64** supports the 64-pin DIP (plastic or ceramic) version of the **87C451**, and features a 3M Textool ZIF socket.

A 40-pin extension is provided with each adapter to allow access to the EPROM programmer's ZIF lever if the adapter is in the way. The **PA451-68** sells for \$165 and the **PA451-64** sells for \$125.

Logical Systems Corporation
6184 Teall Station
Syracuse, NY 13217
(315) 478-0722 • Fax: (315) 475-8460

Reader Service #218

NEWPRODUCTNEWS NEWPRODUCTNEWS

AUDIO DATA ACQUISITION SYSTEMS

Atlanta Signal Processors Inc. has recently introduced two new linear A/D and D/A data acquisition products: the **Serial Voice Interface (SVI)** and the **Serial Audio Interface (SAI)**. Both systems, which are located outside of the computer enclosure and are equipped with a separate power supply, can be connected to virtually any digital signal processing board. The external connection provides a level of performance, noise immunity, and accuracy that is unobtainable with internal systems. It also allows the data acquisition system to be located near the analog source.

The Serial Voice Interface is intended for general-purpose speech (3003400 Hz) processing input and output applications such as speech analysis, speech recognition systems, voice mail systems, telephone line testing, DTMF generation, speech coding, and modem development. The SVI provides 14-bit sampling resolution. It has an on-board switched-capacitor antialiasing input filter and an output reconstruction filter, programmable gain control, telephone handset interface, line-level audio input and output, and front-panel LED status indicators. An optional FCC-registered telephone line interface allows connection to a phone line through an RJ-11 connector.

The Serial Audio Interface is a general-purpose audio band (20-20k Hz) two-channel interface for signal processing input and output applications such as digital audio, voice recognition systems, sonar, and speech coding. The SAI features two channels of input and output and provides 16-bit sampling resolution. It has on-board input antialiasing filters with 64 times oversampling and output-reconstruction filters

with eight times oversampling. Line-level audio input and output, and front-panel LED indicators are also provided.

Both units can be connected to virtually any Digital Signal Processing (DSP) board that has a DSP processor with a serial port. Since these boards typically provide only TTL-level signals, a serial RS-422/TTL adapter is provided to convert to



RS-422 levels. This allows either unit to be located up to 200 feet from the computer.

The Serial Voice Interface is priced at \$695. The FCC-registered telephone line interface is an additional \$200. The Serial Audio Interface costs \$995.

Atlanta Signal Processors, Inc.
770 Spring Street
Atlanta, GA 30308
(404) 892- 7265
Fax: (404) 892-2512

Reader Service #219

We got you covered. . .

... with complete embedded system support for popular PC compilers on Intel 80x 86 and NEC V-Series microprocessors. Now you can develop an embedded application with your choice of compiler—with full support for source level debuggers and in-circuit emulators.

Paradigm LOCATE supports popular C, C++, Pascal, BASIC & Modula-2 compilers from Microsoft, Borland and others.

- comprehensive user's manual
- compiler startup code & examples
- free technical support
- mathcoprocessor emulation support
- extensible MS-DOS emulator
- full Intel OMF output

Lack of an emulator got you down? Not a problem with our Turbo Debugger interface option. Now you can debug your target hardware from the comfort of your PC.

Paradigm LOCATE \$295.00
Turbo Debugger Interface \$195.00

Satisfaction Guaranteed
Orders: (800) 537-5043
Technical Support: (607) 748-5966
BIX: join paradigm
Visa/MasterCard/C.O.D. Accepted

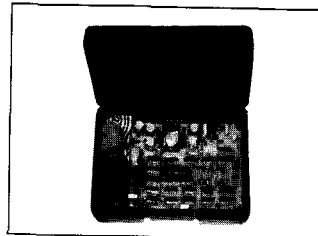


Paradigm Systems
3301 Country Club Road, Suite 2214
Endwell, New York 13760
Turbo Debugger is a registered trademark of Borland International

Reader Service #170

MS-DOS EPROM PROGRAMMING SYSTEM NEEDS NO INTERNAL CARD

EPROMS
2708 (3 supply)
2758, 2716
27C16, 2516
2532*, 2564*
68764*, 68766*
2732, 2732A
27C32, 2764
2764A, 27C64
27128, 27128A
27C128, 27256
27C256, 27512
27C512, 27C010
27010*, 27C1001*



EEPROMS
2804, 2816A
2864A, 28256*

MicroControllers
8741A*, 8742*
8748*, 8748H*
8749*, 8749H*
8751*, 87C51*
8752*, 8744H*

*Socket Adapter Required (Diagrams Included)

CONNECTS TO YOUR SYSTEM'S

PARALLEL PRINTER INTERFACE
A FEATURED TASK TO THE SYSTEM WORKS WITH ANY DESKTOP OR LAPTOP
ADAPTIVE, HIGH-SPEED ALGORITHM MINIMIZE PROGRAMMING TIME AND INSURES VALID DATA
SYSTEM PROGRAM ALL OPERATIONS FROM ANY MANUFACTURER
ALL SYSTEM COMPONENTS FIT NEATLY INTO CASE FOR TRAVEL OR STORAGE

SYSTEM SOFTWARE COMMANDS

- PROGRAM EPROM(S) FROM DISK FILE
 - READ DISK FILE INTO BUFFER
 - SAVE EPROM(S) TO DISK
 - VERIFY EPROM(S)
 - ERASE EPROM(S)
 - COMPARE EPROM(S) WITH BUFFER
 - SELECT DEVICE TYPE
 - DEVICE CHECKSUM
- BUFFER EDITOR** HAS 18 BYTE LEVEL COMMANDS FOR DETAILED OPERATIONS

SYSTEM INCLUDES: PROGRAMMING UNIT, POWER PACK, CONNECTING CABLE, OPERATION **MANUAL & SOFTWARE** **\$239**

SOFTWARE AVAILABLE ON 3 1/2" OR 5 1/4" DISK
TO ORDER SEND CHECK, MONEY ORDER, WRITE OR CALL

VISA

ANDRATECH
P.O. BOX 222
MILFORD, OHIO 45150
(513) 831-9708

MASTER CARD

CALL OR WRITE FOR MORE INFO. -- ADD \$5.00 FOR SHIPPING - \$4.00 COD

Reader Service #106

NEWPRODUCTNEWSNEWPRODUCTNEWS

DATA ACQUISITION BOARD FOR IBM PC

A high-performance 12-bit data acquisition and control board for the IBM PC/XT/AT bus has been announced by Real Time Devices Inc. The ADA2000 features 12-bit A/D conversion, 12-bit D/A conversion, 5-MHz counting, and digital I/O functions. It supports eight channels of differential or 16 channels of single-ended analog input. The A/D converter is a 20-microsecond version and includes a sample-and-hold function that ensures accurate digitization of dynamic signals. The board supports selectable input voltage ranges of 0 to +5, -5 to +5, 0 to 10, and -10 to +10 volts. A programmable-gain amplifier provides software selectable gains of 2, 4, 8, and 16.

Two fast-settling 12-bit D/A converters (which are

closely matched) that have selectable unipolar/bipolar operation and an output range of 5 or 10 volts are also provided. Timing and counting functions are provided by three 5-MHz timer/counters based on the 8253 chip.

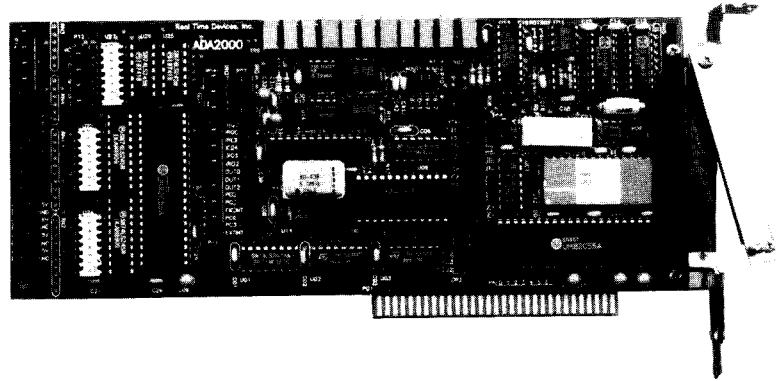
The ADA2000 also includes 40 digital I/O lines. Twenty-four of these lines can be buffered to provide greater drive capability. A unique feature of the board is its ability to route all 120 analog, digital, ground, and power signals through a single PC slot by means of a special clamping bracket. This saves valuable slot space while providing full access to all signals.

Included with each ADA2000 is a disk with sample programs in GWBASIC, Turbo Pascal, Turbo C, and Fortran illustrating control of all the board functions. The ADA2000 is also compatible with the Real Time Devices Atlantis series of data acquisition software.

The ADA2000 sells for \$589.00 in single quantities.

Real Time Devices, Inc.
820 N. University Drive
State College, PA 16804
(814) 234-8087
Fax: (814) 234-6864

Reader Service #220



TALK TO YOUR COMPUTER

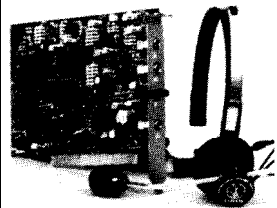
WITH VOICE MASTER KEY®

A PROFESSIONAL VOICE PROCESSING PRODUCT

ADD UP TO 1024 VOICE COMMANDS TO EXISTING PROGRAMS! Speed: data entry and command input to CAD, desk-top publishing, word processing spread sheet, data base, or game programs. Simply train the computer to recognize a word or phrase and assign a series of keystrokes to that command. Pop-up TSP program features pull-down menus and mouse support. Requires under 10K of main memory if EMS present. Near instant response time and high recognition accuracy.

SOUND RECORDING & PLAYBACK
Digitally record your own speech, sound, or music. Software controlled sampling rate (up to 20Khz), graphics-based editing, and data compression utilities. Create software sound files, voice memos, more. Send voice mail through LANs, or modem. DMA data transfer provides continuous recording and playback of sound to hard disk. PC internal speaker supported.

INTERACTIVE SPEECH INPUT/OUTPUT
Tag your own digitized speech files to voice recognition macros. Provides speech response to your spoken commands -- all from within your application software! Make your software come alive!



COMPATIBLE with talking software from IBM, First Byte, Davidson, Optimur Resources, Britannica Software, Electronic Arts, Hyperplot, and many others.

EVERYTHING INCLUDED Voice Master Key System consists of a short plug-in card, durable lightweight microphone headset, software, and manual. Card fits an available slot in your PC or compatible (not for micro channel). Made in the U.S.A.

ONLY \$189.95 (plus shipping & handling)

ORDER HOTLINE: (503) 342-1271 Monday-Friday 8 AM to 5 PM Pacific Time / VISA/MasterCard phone or FAX orders accepted. No CODs. Personal check: subject to 3 week shipping delay. Specify computer type and disk format (3 1/2" or 5 1/4") when ordering. Add \$5 shipping charge for delivery in USA and Canada. Foreign inquiries contact Covox for C & F quotes.

30 DAY MONEY BACK GUARANTEE IF NOT COMPLETELY SATISFIED.

CALL OR WRITE FOR FREE PRODUCT CATALOG.



COVOX INC.

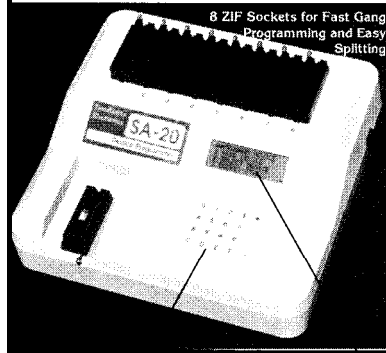
675 CONGER ST.
EUGENE, OR 97402

TEL: (503) 342-1 271
FAX: (503) 342-1 283

EPROM PROGRAMMERS

Stand-Alone Gang Programmer

\$750.00



8 ZIF Sockets for Fast Gang Programming and Easy Splitting

- Completely stand-alone or PC driven Programs E(P)ROMs
- 1 Megabit of DRAM
- User upgradable to 32 Megabit
- 3.6" ZIF socket 8S-232, Parallel In and Out
- 32K internal Flash EEPROM for easy firmware upgrades
- Quick Pulse Algorithm (27256 in 5 sec, 1 Megabit in 17 sec.)
- 2 year warranty
- Made in USA
- Technical support by phone
- Complete manual and schematic
- Single Socket Programmer also available. \$550.00
- Split and Shuffle 16 & 32 bit
- 100 User Definable Macros 10 User Definable Configurations
- Intelligent Identifier
- Binary, Intel Hex, and Motorola S

20 Key Tactile Keypad (not membrane)

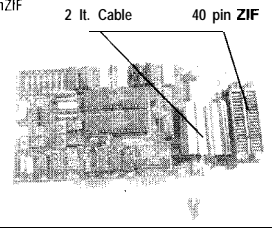
20 x 4 Line LCD Display

Internal Programmer for PC

\$139.95

New Intelligent Averaging Algorithm Programs 64A in 10 sec., 256 in 1 min., 1Meg (27010, 011) in 2 min 45 sec., 2Meg (27C2001) in 5 min. Internal card with external 40 pin ZIF

- Reads, verifies, and programs 2716, 32, 32A, 64, 64A, 128, 128A, 256, 512, 513, 010, 011, 301, 27C2001, MCM 66764.2532
- Automatically sets programming voltage
- Load and save buffer to disk
- Binary, Intel Hex, and Motorola S formats
- Upgradable to 32 Meg EPROMs
- No personality modules required
- 1 year warranty • 10 day money back guarantee
- Adapters available for 8746, 49, 51, 751, 52, 55, TMS 7742, 27210, 57C1024, and memory cards
- Made in USA



NEEDHAM'S ELECTRONICS

4539 Orange Grove Ave • Sacramento, CA 95641
Mon-Fri 9am - 5pm PST

Call for more information

(916) 924-8037

FAX (916) 972-9960

C.O.D.

Reader Service #128

Reader Service #165

NEWPRODUCTNEWS/NEWPRODUCTNEWS

MITSUBISHI 16-BIT FORTH DEVELOPMENT SYSTEM

Home Electronics has announced a combination low-cost FORTH development system/target board for the Mitsubishi 7700 family of 16-bit CMOS single-chip microprocessors.

The development system is a two-board set consisting of a target board that can be used in a stand-alone mode as a single-chip unit with a FORTH kernel, up to 32K bytes of on-chip EPROM, 2K battery-backed RAM, 68 I/O lines, and two RS-232 or RS-422 serial ports, or with a memory expansion board with either 32K of 8-bit RAM or 64K of 16-bit RAM/ROM. The RAM on the expansion boards is also battery backed. The target board has two 40-pin headers for I/O or expansion and a battery backup system with on-board battery. It is 4.5" square without prototyping area or 4.5" by 6" with prototyping area.

The 64K Memory Expansion Board has a very flexible memory mapping scheme that allows splitting the RAM/ROM boundary with a 100-byte resolution so that maximum RAM is available for code development. The FORTH kernel only takes about 6K bytes of ROM, so 58K bytes of RAM are available with this system. The 64K Memory Expansion Board is 4.5" by 6" and piggybacks onto the target board. Also provided on this board are eight chip-select lines for off-board memory or I/O expansion. They can also be used as oscilloscope sync lines for hardware debugging.

The 32K 8-bit memory expansion board is only 1.5" by 2.75" and is oriented so that it doesn't increase the footprint of the target board when it is plugged onto it.

A single supply of +5 V regulated or 8-15 V AC or DC at 30 mA is required. Sockets for up to eight ULN2803s (each having eight 0.5-A current sinks) are on the target board.

Additionally, the 7700 family can address up to 16 megabytes, and has eight 16-bit timers, a watchdog timer, 68 I/O lines, two UARTs, hardware multiply and divide, 19 interrupts, interrupt prioritization, and an 8-bit A/D converter with an 8-channel multiplexer, all with a typical power dissipation of 30 mW. They are available in 8-MHz and 16-MHz versions with 512 to 2K bytes of on-chip RAM and up to 32K of on-chip ROM, 32K EPROM with or without a FORTH kernel, or ROMless.

Also available is a very low cost EPROM programmer board (\$125 with source code) and an EPROM adapter (\$90) to allow programming the on-chip EPROM versions. Additional development tools are available, with source code included, that provide a hex screen editor, macro assembler, both a high-level FORTH word disassembler and an object code disassembler, as well as other utilities. Program development is done on your host PC and downloaded to the target system.

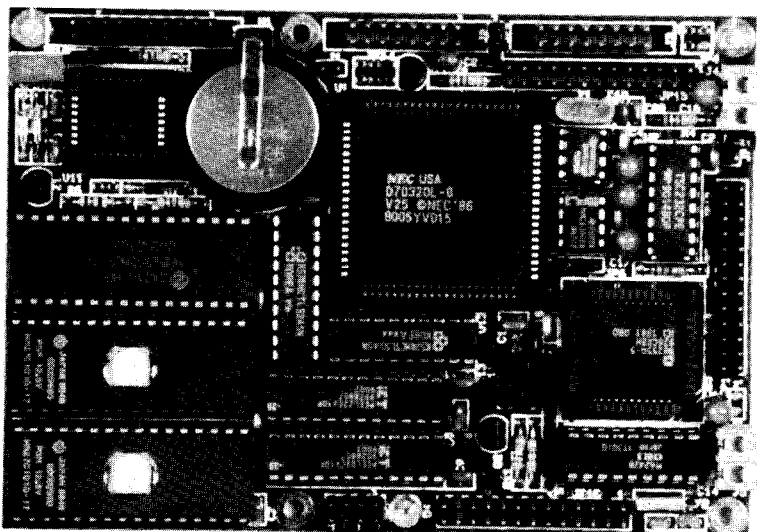
Prices start at \$200 for the 8-MHz ROMless target board and \$60 for the 8-MHz 32K RAM memory expansion board or \$250 for the 8-MHz 16-bit 64K RAM/ROM expansion board with the FORTH kernel in ROM. Quantity discounts are available.

Home Electronics, Inc.
33122 181st Ave. S.E.
Auburn, WA 98002
(206) 735-0790

Reader Service #221

V25 Power Comes to Embedded Control!

Micomint's new RTCV25 is the perfect marriage of a PC-compatible processor, programming convenience, and control I/O. The heart of the RTCV25 is the NEC V25 microprocessor, an all-CMOS, 8088-compatible device running at 8 MHz. The 3.5" x 5" V25 offers engineers 16-bit processing power, large address space, and compatibility with many of the most popular and useful software development tools available today. The RTCV25 enhances the V25's power with 40 parallel I/O lines; a-channel, 8-bit A/D conversion; two serial ports (One RS-232 and one RS-232/RS/485); up to 384K RAM and EPROM; a battery-backed clock/calendar; 1K bit EEPROM, ROM monitor, and the RTC stacking bus. The RTCV25 is compatible with the full line of RTC peripheral boards and products.



Features

- 8MHz V25 processor
- 2 Serial ports
- 40 Parallel I/O lines
- a-channel, 8-bit ADC
- RTC Stacking Bus
- Small 3.5" x 5" size
- 5-volt only operation

Options

- 128 bytes EEPROM
- Battery-backed Clock
- 384K RAM and EPROM
- 8-channel, 10-bit ADC
- ROM Monitor

100 Quantity
OEM Configuration

\$279.00



Actual size
3.5" x 5"

MICROMINT, INC.

4 Park Street
Vernon, CT 06066
call 1-800-635-3355
(203) 871-6170
Fax: (203) 872-2204

NEWPRODUCTNEWS

UNIVERSAL RiTC CUBE

Integrated Vessel Information Corporation has announced the introduction of the Universal RiTC Cube, an enclosure designed specifically for use with the Micromint RTC family of microcontrollers and peripherals. The RiTC cube is made of 16-gauge black anodized aluminum, with a footprint of 5" x 6" and a maximum height of 5".

The RiTC Cube is designed with a "convertible" chassis that allows the height of the Cube to be varied, accommodating RTC stacks of various sizes. At full height, the RiTC Cube has a capacity of seven stacked RTC boards; shortened to 3", the RiTC holds a stack of three RTC boards. In both cases, the RiTC can contain an RS05 power supply in addition to the controllers and peripherals.

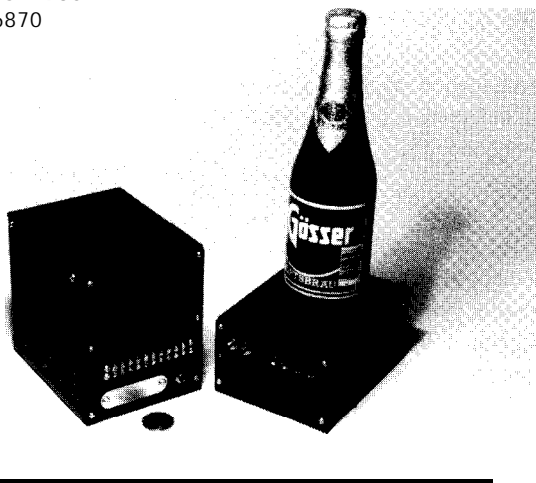
Full-size gridded layout templates are provided for designers to layout and install custom hardware cutouts in the faceplates. RiTC Cubes are assembled with machine screws into PEM nuts to allow repeated assembly/disassembly cycles during development or servicing.

RiTC Cubes are priced beginning at \$99.95, quantity one.

Integrated Vessel Information Corporation

671 Via Alondra, Unit 805
Camarillo, CA 93012
(805) 389-6870

Reader Service #222



96-LINE I/O

A 96-line digital I/O board for the IBM PC Bus has been introduced by Centrum Research. The CT-70009 consists of four Intel 8255 Programmable Peripheral Interface chips to provide unidirectional and bidirectional strobed I/O. Eight interrupts are available for the 8255 chips. The interrupts are jumper-selectable for IREQ2-IREQ7. DIP switches provide address selections for the ports.

The CR-70009 is equipped with four 50-pin headers for use with industry-standard PB8, PB16, or

PB24 optoisolated I/O module backplanes. Each header is positioned so that cabling can easily be placed through the card's bracket. Anticipated applications for the CR-70009 include event sensing, process control, relay activation, and security.

The CR-70009 has an introductory price of \$149.00, quantity one.

Centrum Research, Inc.
45 14 Cole Avenue
Suite 600
Dallas, TX 75205
(214) 559-7175

Reader Service #223

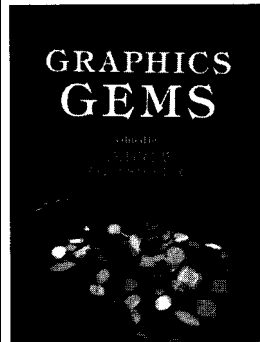
GET GRAPHIC

Graphics Gems

edited by
Andrew S. Glassner

This handbook provides practical solutions to graphics problems, and every graphics programmer will find it an essential tool in saving time and energy in their daily programming activities.

August 1990, 880 pp., \$49.95
ISBN: 0-12-286165-5

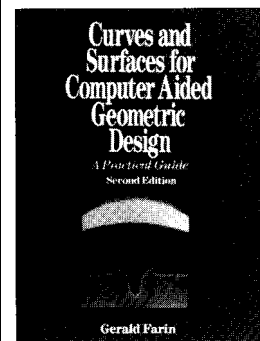


Curves and Surfaces for Computer Aided Geometric Design

A Practical Guide
SECOND EDITION

Gerald Farin

May 1990, 464 pp., \$39.95
ISBN: 0-12-249051-7



The Desktop Fractal Design System

Michael F. Bamsley

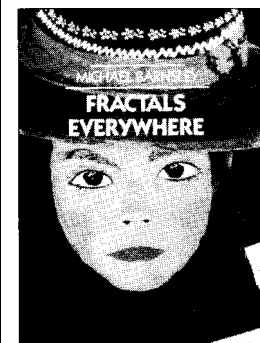
Includes The Desktop Fractal Design Handbook and one floppy disk.

IBM Version: The system requires an IBM, or compatible, PC with a graphics board (EGA or VGA) and 640K memory.

1989, \$39.95/ISBN: 0-12-079063-7

Macintosh version: The system runs on Macintosh Plus, the Macintosh SE series, and the Macintosh II family of computers, with a megabyte of memory. Color graphics is not required. No math coprocessor is necessary. The software will work with version 6.0 or higher of the Macintosh operating system.

August 1990, \$39.95 (tentative)
ISBN: 0-12-079064-5



Fractals Everywhere

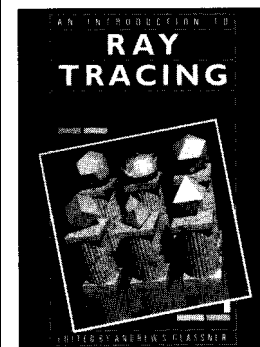
Michael F. Bamsley

1988, 394 pp., \$39.95/ISBN: 0-12-079062-9

An Introduction to Ray Tracing

edited by
Andrew S. Glassner

1989, 327 pp., \$49.95/ISBN: 0-12-286160-4



Write for a FREE brochure!

Order from your local bookseller or directly from

ACADEMIC PRESS



Harcourt Brace Jovanovich, Publishers
Book Marketing Department #15100
1250 Sixth Avenue, San Diego, CA 92101

CALL TOLL FREE
1-800-321-5088

Quote this reference number for free postage and handling on your prepaid order #15100

Prices subject to change without notice ©1990 by Academic Press, Inc. All Rights Reserved. TC/MJD—15100

Reader Service # 101

October/November 1990 15

FEATURE ARTICLE

Saim Ural

Functions of Complex Variables

Generating Biomorphs on Personal Computers

Over the last five years, I am sure you have seen some magnificent, computer-generated color pictures of fractals. Some of these pictures are generated by iterating a function with complex variables. The first image in Figure 1 is created using $Z^3 + C$ as the function, and the second image is created using $\sin(C \times Z) + Z^2 + C$ where Z is a complex variable and C is a complex constant. For programmers, creation of these images poses numerous challenges.

Other than performing the iterations and displaying the image, a programmer now has an extra challenge: How to evaluate the functions $Z^3 + C$ or $\sin(Z)$ or Z^2 , and so on. Most of the high-level languages that are commonly available to the programmer these days, with the exception of good old FORTRAN, have forgotten about complex numbers. Compilers supporting most of the high-level languages do not support the complex type anymore.

In this article, I will show you how to perform basic operations on complex numbers and will define algorithms with which various functions, such as $\sqrt{}$, \exp , \ln , and others, of a complex variable can be calculated. I will also put these functions into use and show you how images like the ones included in this article can be created. I wrote several procedures to calculate these functions of complex variables in both Turbo Pascal and Modula-2, but **they can easily** be translated to any other high-level language, such as C or Ada. [Editor's Note: Software for this article is available for downloading or on Software On Disk #17. See page 108 for downloading and ordering information.]

First a word or two on complex variables. A complex value is represented as:

$$z = a + i \times b$$

where a and b are real numbers, and i is the square root of negative one. Variable a is called the real part of the complex number, and b is called the imaginary part. Although mathematicians use the letter i to represent the square root of negative one, people in the field of electronics like to use j for the same purpose (since i commonly denotes current), so you might have seen this number written as:

$$z = a + j \times b$$

One can visualize a complex value as a point in Cartesian coordinate system where the axes are labeled as real and imaginary.

DATA ABSTRACTION

In a high-level language environment, we first should decide on a data structure to represent complex numbers since such a type is not available as a standard type. The obvious choice for this is a record structure. I will use

the following record to represent complex variables:

```
Complex = RECORD
    Re,
    Im : REAL
END; { RECORD }
```

This representation will allow us to declare variables of complex type without worrying about its components.

In teaching computer science, we always emphasize the importance of data abstraction. This means that a user wishing to perform operations on the type we have defined should be able to do so without paying attention to the actual representation of that type, maybe (and preferably), without knowing anything about that representation. With this idea in mind, creating a data structure requires the programmer to create a set of procedures to accommodate the needs of any user. In the case of complex type abstraction, we have to provide input and output procedures, assignment and initialization procedures, and, of course, procedures to perform the basic arithmetic operations as well as transcendental and trigonometric functions.

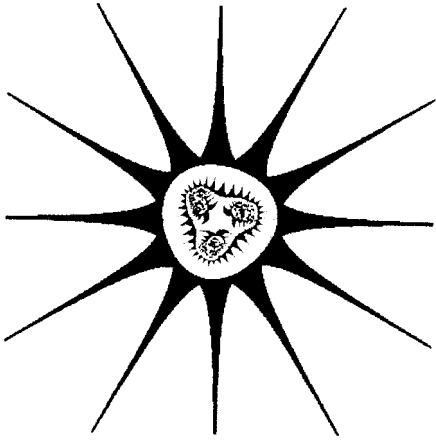


Figure 1—These biomorphs are good examples of what can be created using the procedures described in the text.

INPUT AND OUTPUT

Since every language provides a way of reading and printing a floating-point (real) number, writing procedures to read and write a complex variable can easily be achieved. For example, knowing that reading a complex value requires one to read both the real and the imaginary parts, a `ReadComplex` procedure can be written as

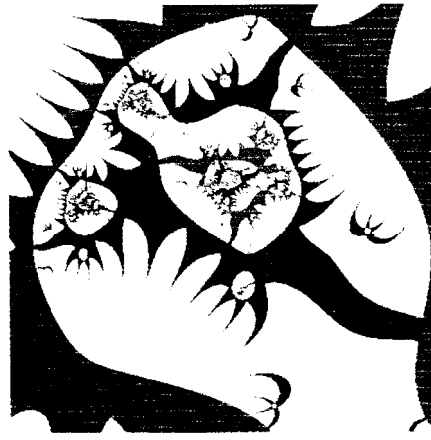
```
PROCEDURE ReadComplex (VAR
    Out:Complex);
BEGIN
    Read (Out.Re);
    Write (' ');
    Read (Out.Im);
END; { ReadComplex }
```

The other input and output procedures are given in Listing 1. Note that other than a `WriteComplex` procedure, two other procedures to write the real and imaginary parts of a complex number are also included. This is important when we create a new data type.

INITIALIZATION AND ASSIGNMENT

A user of complex variables needs to create complex variables and needs to copy the value of one variable to the other. These two operations can be performed by calling the `CAssign` and `CCopy` procedures. For example, if we need to create a complex variable that is equal to $(5.0 + 3.2i)$, then we have to call `CAssign` as

```
CAssign (5.0, 3.2, Z);
```



Similarly, `CCopy` can be used to copy a variable into another.

BASIC ARITHMETIC OPERATIONS

One of the basic operations is the complement. The complement of a complex variable $(a + ib)$ is defined as $(a - ib)$; the procedure `CComplement` will be used to perform this operation.

Addition and subtraction of complex variables can be accomplished by adding or subtracting the real and imaginary parts respectively, that is

$$(a + ib) \pm (c + id) = (a + c) \pm (b + d)i$$

The multiplication of two complex variables requires a bit more work. In this case we have to remember that $i \times i = -1$. Keeping this in mind,

$$(a + ib) \times (c + id) = (ac - bd) + (ad + bc)i$$

The division operation, $(a + ib) / (c + id)$, is accomplished by first multiply-



Figure 2—These biomorphs were created using the equations $f(z) = \sin(z) + \exp(z) + C$ and $f(z) = z^p + z + C$ and the parameters shown in Table 1a.

ing both sides of the fraction by the complement of $(c + id)$. This leaves a real number in the denominator. Dividing the real and the imaginary parts of the nominator by this real number gives us the required result.

```
FUNCTION CDiv (P1, P2:Complex; VAR
    P:Complex);
{ P := P1 / P2 }
BEGIN
    (Get complement of denom)
    CComplement (P2, P);
    {Find compl times denom}
    CMult (P2, P, P2);
    {Find compl times nomin}
    CMult (P1, P, P1);
    (Find resulting imaginary)
    P.Re := P1.Re / P2.Re;
    P.Im := P1.Im / P2.Re;
END; { CDiv }
```

The procedures for the complement and the four arithmetic operations are given in Listing 2.

The algorithms to calculate \sqrt{z} , z^n , z^p , $\ln(z)$, $\exp(z)$, $\sin(z)$, and $\cos(z)$, where z and p are complex variables and n is an integer, will be given below. I will assume that the programming language we are using enables us to compute \sin , \cos , \ln , and \exp of real variables. Using the data abstraction described above and the upcoming functions, one can generate images similar to the ones shown in Figure 1. The algorithms to create these images, called biomorphs, will be explained a bit later.

NONTRIGONOMETRIC FUNCTIONS

Let us start with the easiest function to compute: the absolute value.



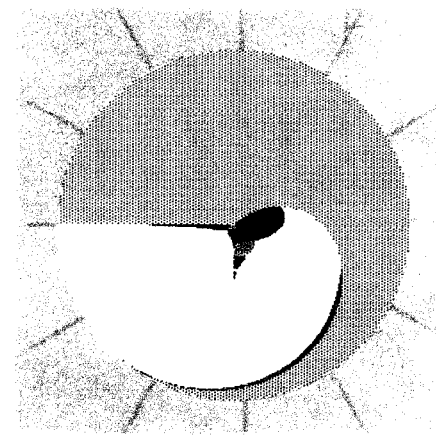
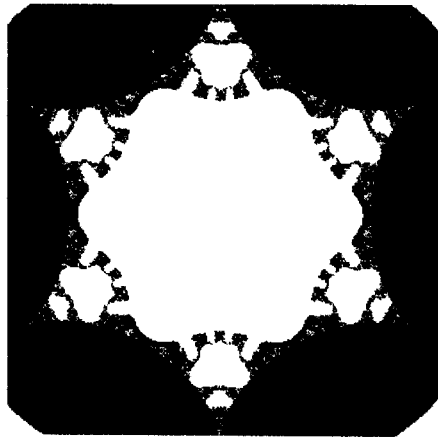
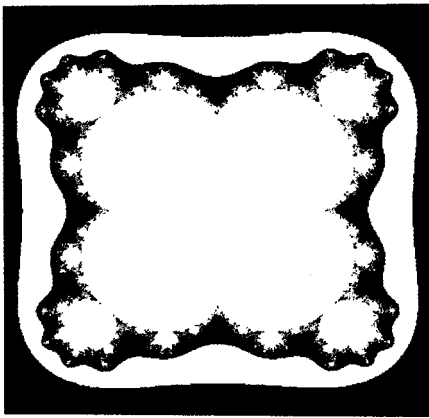


Figure 3—These images show portions of quasi-Mandelbrot sets where $p = 5, -5,$ and i . Other parameters used are in Table 1b.

This function can easily be constructed if we remember the definition of the absolute value of a complex variable:

$$|a + ib| = \sqrt{a^2 + b^2}$$

A function `CABs` can be written as:

```
FUNCTION CABs(A:Complex):REAL;
{ Returns |A| }
BEGIN
  CABs := sqrt(A.Re*A.Re +
              A.Im*A.Im)
END; { CABs }
```

The calculation of the square root of a complex value is more complicated. If the square root of a complex number is represented as $(c + id)$, then the square root of a complex number $(a + ib)$ is found by using the following algorithm:

```
If both a and b are zero Then
  both c and d are zero
Else
  If a ≥ 0 Then
    c = (sqrt(|a| + |a+ib|)) / 2
    d = b / 2c
  Else
    d = sign(b) * (sqrt(|a| +
                      |a+ib|)) / 2
    c = b / 2d
```

If n is an integer, then the easiest way to compute z^n is to multiply z by itself n times. Although this algorithm is not the most efficient way of accomplishing the task, it is simple and can easily be accomplished by calling the multiplication procedure shown in Listing 2.

Calculating z^p where p is either a real number or a complex number can only be done by first taking the loga-

rithm of z^p which is equal to $[p \times \ln(z)]$. If we can compute $\ln(z)$, then this product can be found. Let us call this product f . Then z^p will be equal to $\exp(f)$ where f may be another complex number. Therefore, calculation of z^p needs the calculation of $\ln(z)$ and $\exp(z)$. Let's see how these two functions of a complex value can be found.

The value of e^z where z is a complex number of the form $(a + ib)$ can be found by using the following identity:

$$e^z = e^{a+ib} = e^a e^{ib} = e^a (\cos(b) + ix \sin(b))$$

The evaluation of $\ln(z)$ is not as simple as e^z . In this case, the real part of the result is equal to $\ln(|z|)$, whereas the imaginary part is given as $\arctan(b/a)$.

If we have these two functions of complex variables available, then we can compute z^p as:

$$z^p = \exp[p \times \ln(z)]$$

The procedures to compute the `sqrt`, `ln`, and `exp` of a complex variable are given in Listing 3. The function procedure `FindAngle` enables us to compute `arctan(b/a)` without creating divide-by-zero error even if a is zero. The value returned will be in the range from $-\pi$ to $+\pi$.

TRIGONOMETRIC FUNCTIONS

There are two more functions of complex variables we need to know how to compute: sine and cosine.

```
PROCEDURE ReadComplex(VAR Out:Complex);
BEGIN
  Read(Out.Re);
  Write(' ');
  Read(Out.Im)
END; { ReadComplex }

PROCEDURE WriteComplex(P:Complex; N,D:INTEGER);
{ Write a complex number, both the real and the imaginary
  parts, using N spaces, D of which will be after the
  decimal point. }
BEGIN
  Write('(',P.Re:N:D,', ',P.Im:N:D,' I)')
END; { WriteComplex }

PROCEDURE WriteRealPart(P:Complex; N,D:INTEGER);
{ Write the real part of a complex number using N spaces,
  D of which will be after the decimal point. }
BEGIN
  Write(P.Re:N:D)
END; { WriteRealPart }

PROCEDURE WriteImagPart(P:Complex; N,D:INTEGER);
{ Write the imaginary part of a complex number using N
  spaces, D of which will be after the decimal point. }
BEGIN
  Write(P.Im:N:D)
END; { WriteImagPart }
```

listing 1 -Normal I/O procedures must be rewritten with complex variables in mind.

```

PROCEDURE CComplement (P1:Complex; VAR P2:Complex);
{ P2 is complement of P1 }
BEGIN
  P2.Re := P1.Re;
  P2.Im := -P1.Im
END; { CComplement }

PROCEDURE CAdd (P1,P2:Complex; VAR P:Complex);
{ P := P1 + P2 }
BEGIN
  P.Re := P1.Re + P2.Re;
  P.Im := P1.Im + P2.Im
END; { CAdd }

PROCEDURE CSubt (P1,P2:Complex; VAR P:Complex);
{ P := P1 - P2 }
BEGIN
  P.Re := P1.Re - P2.Re;
  P.Im := P1.Im - P2.Im
END; { CSubt }

PROCEDURE CMult (P1,P2:Complex; VAR P:Complex);
{ P := P1 * P2 }
BEGIN
  P.Re := P1.Re * P2.Re - P1.Im * P2.Im;
  P.Im := P1.Re * P2.Im + P1.Im * P2.Re
END; { CMult }

PROCEDURE CDiv (P1,P2:Complex; VAR P:Complex);
{ P := P1 / P2 }
BEGIN
  CComplement (P2,P);
  CMult (P,P,P2);
  CMult (P1,P,P);
  P.Re := P1.Re / P2.Re;
  P.Im := P1.Im / P2.Re
END; { CDiv }

```

listing 2-Procedures to implement ordinary arithmetic are no more than a few lines long.

Quite similar to the algorithm of the exp function, we need to refer to the definition of these two functions. The sine of a complex number is:

$$\begin{aligned} \sin(z) &= \sin(a + ib) \\ &= \sin(a) \times \cosh(b) \\ &\quad + i \times \cos(a) \times \sinh(b) \end{aligned}$$

whereas the cosine of a complex value is defined as:

$$\begin{aligned} \cos(z) &= \cos(a + ib) \\ &= \cos(a) \times \cosh(b) \\ &\quad - i \times \sin(a) \times \sinh(b) \end{aligned}$$

Unfortunately, these two functions require us to deal with the sinh and cosh functions that are not usually available in most of the programming language libraries with the exception of FORTRAN. Although there are a number of methods of computing these two functions, like using their series approximation, I would like to describe algorithms that are much faster and highly accurate. The sinh(A), where A is a real number, can be found using the algorithm given in

the function shown in Listing 4. Note that the equations to be used depend on the value of A, and the equation for the case where $A < 0.3465$ is carefully written to reduce the number of multiplications needed.

The value of cosh(A) can be computed using the formula

$$\cosh = (W + 1.0/W) / 2.0$$

where $W = \exp(|A|)$. The procedures for sinh(A), cosh(A), sin(z), and cos(z) are given in Listing 4.

I used these procedures to obtain the images of Figure 1. In the next section we will discuss the algorithms with which the images in Figures 2 and 3 were created. Figure 2 contains pictures of biomorphs, while Figure 3 shows the pictures of quasi-Mandelbrot sets of various orders.

The creation of these images on a computer requires familiarity with the complex numbers and their functions. In previous sections, I explained the algorithms for performing a variety of operations on complex numbers. In this section, I will make use of those

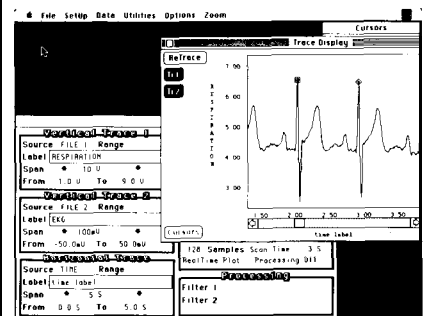
DATA ACQUISITION SOFTWARE

We'll never
leave you
without
a trace

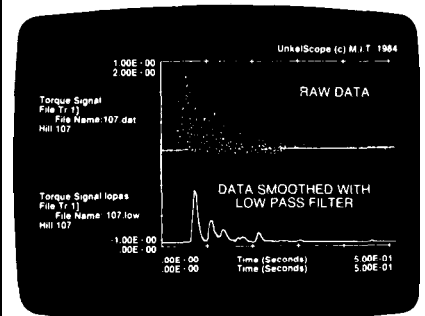
UnkelScope[®]

UnkelScope is an easy-to-use, menu-driven software package that will always leave you with a clear, accurate trace. Whether you're in a laboratory or on an oil rig in the North Atlantic, UnkelScope will get the job done

- Full hardware speed
- Real-time X-Y plots
- Graphical Editing
- Data Processing
- Experiment Control
- Plus much more



MAC Version



PC Version

UnkelScope JUNIOR \$125
UnkelScope for MAC \$149
UnkelScope Level 2+ \$549

30-day money-back guarantee

(617) 861-0181

FAX (617) 861-1850

Unkel Software Inc.

62 Bridge Street, Lexington, MA 02173

Reader Service #206

October/November 1990 19

```

PROCEDURE CSqrt(A:Complex; VAR P:Complex);
{ Returns sqrt(A) }
BEGIN
  IF (A.Re=0.0) AND (A.Im=0.0) THEN
    BEGIN
      P.Re := 0.0;
      P.Im := 0.0
    END
  ELSE
    IF A.Re >= 0.0 THEN
      BEGIN
        P.Re := sqrt((ABS(A.Re) + CAbs(A))/2.0);
        P.Im := A.Im / (2.0*P.Re)
      END
    ELSE
      BEGIN
        P.Im := Sign(A.Im) * sqrt((ABS(A.Re)+CAbs(A))/2.0);
        P.Re := A.Im / (2.0*P.Im)
      END { IF }
    END; { CSqrt }

FUNCTION FindAngle(x,y:REAL):REAL;
VAR
  Theta : REAL;
BEGIN
  IF y <> 0.0 THEN
    BEGIN
      Theta := arctan(x/y);
      IF y < 0.0 THEN
        Theta := Theta + Pi * Sign(x)
      END { IF }
    END
  ELSE
    Theta := Sign(x) * Pi / 2.0;
  FindAngle := Theta
END; { FindAngle }

PROCEDURE CExp(A:Complex; VAR P:Complex);
{ Returns exp(A) }
BEGIN
  P.Re := exp(A.Re) * cos(A.Im);
  P.Im := exp(A.Re) * sin(A.Im)
END; { CExp }

PROCEDURE CLn(A:Complex; VAR P:Complex);
{ Returns ln(A) }
BEGIN
  P.Re := ln(CAbs(A));
  P.Im := FindAngle(A.Im,A.Re)
END; { CLn }

PROCEDURE CIntPower(A:Complex; N:INTEGER; VAR P:Complex);
{ Returns A ** n }
VAR
  i : INTEGER;
BEGIN
  CCopy(A,P);
  FOR i := 1 TO N-1 DO
    CMult(A,P,P)
  END; { CIntPower }

PROCEDURE CCompPower(A1,A2:Complex; VAR P:Complex);
{ Returns A1 ** A2 }
VAR
  A : Complex;
BEGIN
  CLn(A1,A);
  CMult(A2,A,A);
  CExp(A,P)
END; { CCompPower }

```

listing J--Power, exponentiation, square root, and logarithm procedures.

functions and explain the algorithms for creating these wonderful images.

BIOMORPHS

Biomorphs were discovered by Clifford A. Pickover around 1985 while experimenting with complex func-

tions. The mathematical principle behind the biomorphism involves iterations in the complex domain and checking the convergence or divergence of these iterations. Let's consider a function of a complex variable:

$$f(z) = z^3 + c$$

where z is a complex variable and C is a complex constant. For a given point z_0 , the iterations of this function are obtained as:

$$z_1 = f(z_0) = (z_0)^3 + c$$

$$z_2 = f(z_1) = (z_1)^3 + c$$

...

$$z_n = f(z_{n-1}) = (z_{n-1})^3 + c$$

where $n=1,2,3,\dots,\infty$. This sequence may converge to a value or may diverge toward infinity depending on z_0 and C . Since it is impossible to perform these iterations an infinite number of times, we would like to use some criteria to decide when to stop iterating on a function. The criteria for creating biomorphs are:

1. Stop the iterations if $|z_{\text{real}}| > \epsilon$ or $|z_{\text{imag}}| > \epsilon$
2. Do the iterations for a fixed number (n) of times.

The pictures in Figure 2 are obtained using $\epsilon = 10$, $n = 10$.

In order to generate images of the biomorphs, we first decide on a number of initial parameters. These are:

1. The complex domain that we would like to work with. This requires selecting a point in the complex domain as one corner and defining the size of the domain in the real and imaginary coordinates. Let us call these z_s , RealSize , and ImagSize .
2. The number of pixels we would like to use along the real and imaginary axes. Assume these are called NReal and NImag .
3. Define a value for the constant C .

If we know the values of z_s , C , RealSize , ImagSize , NReal , and NImag , the algorithm to create these images can be given as:

```

dRe := RealSize / NReal
dIm := ImagSize / NImag
For i := 0 to NReal Do
  For j := 0 to NImag Do
    z := (i*dRe, j*dIm) + z_s
    Iter := 0
    Repeat
      z := f(z,C)
      increment Iter by 1

```

```

Until |zreal| > 10 or
      |zimag| > 10 or
      Iiter > n
If |zreal| < 10 or
  |zimag| < 10 Then
  display pixel i,j as black
Else
  display pixel i,j using
  color Iiter
End If
End For
End For

```

I usually start with large values for the z_s , RealSize, and ImagSize, and use small values for the NReal and NImag. This allows me to see the image of a large domain with fewer computations. If I see an interesting figure, then I try zooming on to that part of the image by arranging the values of z_s , RealSize, and ImagSize. After finding a very interesting image, I use large values for NReal and NImag to create an image that is filling the monitor.

THE MANDELBROT SETS

The Mandelbrot set is named for Benoit B. Mandelbrot, from whose work the field of study of fractal geometry emerged. The algorithm behind the Mandelbrot set is quite similar to the biomorphism algorithm with some differences. First of all the equation to use for this set is

$$f(z) = z^2 + c$$

where z is a complex variable and C is a complex constant. The iterations of this equation produce the Mandelbrot set when the initial values to z_s , RealSize, ImagSize, NReal, and NImag are known. The basic difference is the criteria used to stop the iterations and the number of iterations performed. The number of iterations allowed, NIter, is much higher than the one we used for biomorphs. This time the number of iterations should be between 100 to 1000. We also stop the iterations when $|z| > 2$ because from this point on, the value of z will go toward infinity. With these in mind, the algorithm for the Mandelbrot set can be given as:

```

dRe := RealSize / NReal
dIm := ImagSize / NImag

```

```

FUNCTION Sinh(A:REAL):REAL;
VAR
  W : REAL;
BEGIN
  IF ABS(A) < 0.3465736 THEN
    BEGIN
      W := A * A;
      Sinh := ((0.00836915*W) + 0.16666505*W) + 1.0 * A
    END
  ELSE
    BEGIN
      W := exp(ABS(A));
      Sinh := Sign(A) * (W - 1.0/W) / 2.0
    END { IF }
  END; { Sinh }

```

```

FUNCTION Cosh(A:REAL):REAL;
VAR
  W: REAL;
BEGIN
  W := exp(ABS(A));
  Cosh := (W + 1.0 / W) / 2.0
END; { Cosh }

```

```

PROCEDURE CSin(A:Complex; VAR P:Complex);
{ Returns sin(A) }
BEGIN
  P.Re := sin(A.Re) * Cosh(A.Im);
  P.Im := cos(A.Re) * Sinh(A.Im)
END; { CSin }

```

```

PROCEDURE CCos(A:Complex;VAR P:Complex);
{ Returns cos(A) }
BEGIN
  P.Re := cos(A.Re) * Cosh(A.Im);
  P.Im := -sin(A.Re) * Sinh(A.Im)
END; { CCos }

```

Listing 4—Trigonometric functions require the use of the hyperbolic sine (sinh) and hyperbolic cosine (cosh) functions which aren't often found in standard function libraries.

Position and/or Velocity

Motion Control

Highly integrated, s&m-of-the-art Digital Servo Controller provides
MAXIMUM PERFORMANCE at **MINIMUM COST**

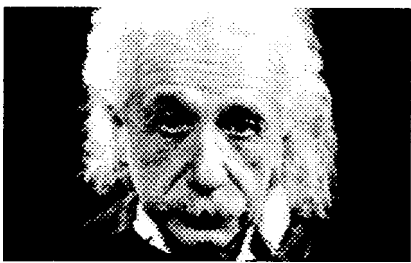


MODEL DC-2

- 2 axes DC servo control with PI D-FF
- Incremental encoder, to 1.0 MHz, with single ended or differential receiver.
- 1.5 amp direct motor drive (to 75 watts) output, and/or **±10volt** analog signal (12 bit DAC) output
- Additional 2 axes stepper control capability
- 32K bytes **ROM**, 32K bytes non-volatile RAM
- Install in PC/XT/AT compatible, or use stand-alone
- 32 bit precision (1 part in 4 billion)
- High level interface libraries in "C" and **BASIC**, with examples and source code, included
- 16 **bit** Microcontroller



Precision Micro Control
 CORPORATION
 3555 Aero Court, San Diego, CA 92123
 (619) 565-1500 FAX (619) 565-1166



Introducing VICTOR, the video capture and image processing library

Victor is a library of functions for C programmers that simplifies development of scientific imaging, quality control, security, and image database software. Victor gives you device control, image processing, display, and TIFF/PCX file handling routines.

Your software can have features such as: live video on VGA, resize and zoom, display multiple images with text and graphics. Image processing functions include brightness, contrast, matrixconvolution filters: sharpen, outline, etc, linearization, equalization, math and logic, overlay, resize, flip, invert, mirror. Size/number of images limited only by memory. Display on EGA/VGA up to 800x600x256. And, to get you up and running quickly, we've included our popular Zip Image Processing software for rapid testing and prototyping of image processing and display functions.

Victor supports Microsoft C, QuickC, and Turbo C . . . all for only \$195.

Victor and Zip support ImageWise and other popular video digitizers.

NEW! ZIP Colorkit, the software that allows any gray scale digitizer to create photographic quality color images.

It's easy to produce stunning color images with the ZIP Colorkit -- capture three images with a gray scale video digitizer using red, green, and blue filters and save the images. Colorkit loads the image files and uses a unique optimizing algorithm to calculate the optimum color image. Supports PIW, PIF, PCX, TIFF, GIF, and TGA file formats. ZIP COLORKIT, \$75.

VICTOR LIBRARY includes FREE
ZIP Image Processing \$195

VICTOR LIBRARY with video
frame grabber \$349

ZIP Colorkit \$75

Call (314) 962-7833 to order
VISA/MC/COD

CATENARY SYSTEMS
470 BELLEVUE
ST LOUIS MO 63119
(314) 962-7833

Equation	z_0	C	RealSize	ImagSize
$\sin(z) + \exp(z) + C$	(-3.0,-3.5)	(-5.5)	5.6	5.01
$z^5 + z^2 + C$	(-3.0,3.0)	(-1,-1)	6.0	6.0

z_0	p	RealSize	ImagSize
(-1.095,-1.274)	(5.0)	2.19375	2.457
(-1.560,-1.560)	(-5.0)	3.06	3.06
(-2.520,-2.650)	(0,i)	5	5

Table 1 — (a) Values used to generate the biomorphs in Figure 2. (b) Values used to make the Mandelbrot sets in Figure 3.

```

For i := 0 to NReal Do
  For j := 0 to NImag Do
    z := (0,0);
    C := (i*dRe, j*dIm) + z;
    Iter := 0
    Repeat
      z := z^2 + c
      increment Iter by 1
    Until |z| > 2 or Iter > n
    If |z| < 2 Then
      display pixel i,j as black
    Else
      display pixel i,j using
        color Iter
    End If
  End For
End For

```

THE QUASI-MANDELNBROT SETS

The quasi-Mandelbrot sets are a variation on the original Mandelbrot set. The only difference in this case is that instead of iterating on $f(z) = z^2 + C$, we will iterate on $f(z) = z^p + C$, where p is another complex number. The set and therefore the images we can obtain from these equations are drastically different from the original Mandelbrot set and can be as intricate as the original one. With the help of the algorithms presented above, it becomes rather easy to perform these iterations.

THE IMAGES

Figure 2 contains images of biomorphs that are obtained using $f(z) = \sin(z) + \exp(z) + C$ and $f(z) = z^5 + z^2 + C$, respectively. The parameters used to obtain these images are listed in Table 1a.

Figure 3 shows the images of quasi-Mandelbrot sets where p was 5, -5, and i , respectively. The values of

the parameters that I used to obtain these images are given in Table 1b.

In order to create these images, I used an IBM PC with an Intel Inboard 386 which also has an Intel 387 math coprocessor. The Intel Inboard was donated to me by Intel

to work on this project. I am grateful to them. The monitor I have is an NEC Multisync color monitor with Vega VGA card. A 400 x 400 image of a biomorph and standard Mandelbrot set usually takes 30 minutes to generate on this computer. The time needed for a quasi-Mandelbrot set of the same size is about 3.5 hours. Of course, if you have a computer more powerful than mine, you will be spending less time to generate these images. The images are printed using an HP PaintJet printer. The complete Turbo Pascal 5.0 program that generates the images of biomorphs can be downloaded from Circuit Cellar BBS.

I have used a number of equations to generate images of biomorphs. Among them I can mention

1. $f(z) = z^2 + C$
2. $f(z) = z^3 + c$
3. $f(z) = z^5 + c$
4. $f(z) = z^2 + z^6 + c$
5. $f(z) = z^5 + z^2 + c$
6. $f(z) = \sin(z) + \exp(z) + C$
7. $f(z) = \sin(z) + z^2 + C$
8. $f(z) = \sin(C \times z) + z^2 + C$

I would like to know what kind of equations you may come up with. Please send your suggestions to me in care of CIRCUIT CELLAR INK. ❖

Saim Ural is an associate professor of Computer Science at Western Washington University. His interests include computer graphics, fractals, and computer security.

IRS

- 250 Very Useful
- 251 Moderately Useful
- 252 Not Useful

FEATURE ARTICLE

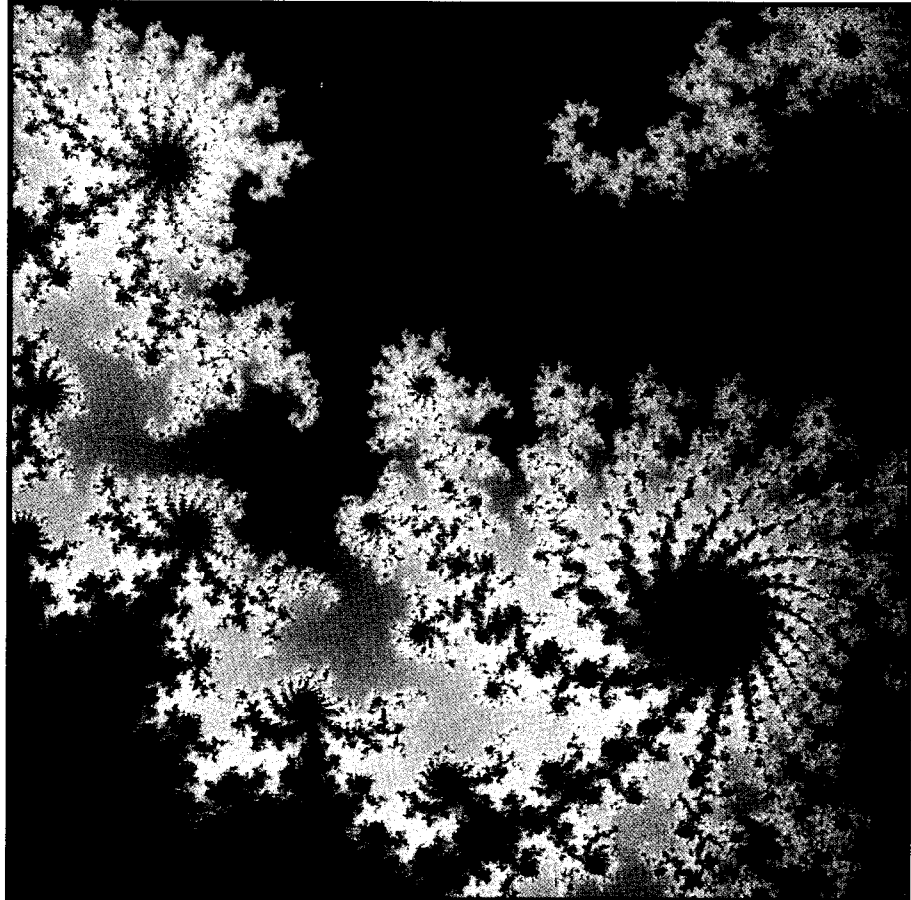
Chris Ciarcia

Creating Fractal Images

Using the Power of Fractals for Realistic Planetary Images

Those of us who dabble in the world of computer art are always looking for ways to make our synthesized pictures as realistic as possible. But unlike the conventional painter who simply draws his interpretation of a scene on canvas, the measure of our success is often based on our ability to create imaginative models of the objects we wish to render. In many ways this represents a degree of difficulty that often exceeds that experienced by the conventional artist. He need only create a personal interpretation of the visual representation of a form; while we often find it necessary to know and understand the fundamental mechanisms from which its ultimate form is derived. He creates images through simple experience and expression, while we create images through experience, "understanding and modeling," and then expression.

Our humble beginnings in computer art began with simple drawings created using straight line segments and polygons, and then evolved into several sophisticated and powerful classes of geometric modeling techniques based on parametric curves, surfaces, and solid-object modeling [1]. However, most of these rendering tools were designed for relatively simple forms. They were highly successful when describing man-made structures or creations, but when it came to natural objects, such as land masses, mountains, trees, clouds, water, fire, and so on, these basic tools were not enough. Especially since most of these natural class objects have irregular or fragmented features which do not lend themselves to simple Euclidean modeling. As a result, new



tools were developed using fractal-geometry methods which were originally proposed by Mandelbrot in his book "The Fractal Geometry of Nature" [2]. These new tools use fractal forms which represent the physical visual characteristics of natural objects and/or phenomena. They are described as geometrical entities that are related to the fundamental processes taking place. For instance, suppose we wanted to create a two-dimensional image of a typical rain cloud. This can be accomplished by generating a stochastic fractal model

which uses as its fundamental geometrical fractal shape a two-dimensional representation of what we know the basic process to be, mainly turbulence characterized by a swirl or eddy of diffuse water vapor. These swirls or eddies can then be scaled and combined in a stochastic fashion to create our desired scene.

FRactal Patterns Arising in Chaotic Dynamical Systems

I would like to begin with the simple dynamical system, $F(z) = z^2 +$

c, where both z and c are defined as complex numbers. That is, $z = z_r + iz_i$ and $c = c_r + ic_i$ with $z_r, z_i, c_r,$ and c_i being real numbers and i is the imaginary constant $(-1)^{0.5}$. Here the subscript r represents the real part and i represents the imaginary part. Since these complex numbers are designated by a set of real numbers, they can be plotted in a Cartesian coordinate system by aligning the real part with the x-coordinate line and the imaginary part with the y-coordinate line. The absolute magnitude of z ,

$$|z| = zm = (z_r^2 + z_i^2)^{0.5} \quad (1)$$

is therefore a measure of the distance from the origin of that Cartesian system to the point z .

Using these definitions, our chosen dynamical system can now be rewritten in the following form:

$$\begin{aligned} F(z) &= z^2 + c \\ &= (z_r + iz_i) \times (z_r + iz_i) + (c_r + ic_i) \\ &= z_r^2 + i^2 z_i^2 + 2iz_r z_i + c_r + ic_i \\ &= (z_r^2 - z_i^2 + c_r) + i(2z_r z_i + c_i) \end{aligned} \quad (2)$$

And we can create a Julia set image by processing the function, $F(z)$, in an iterative fashion for each point in our complex plane defined by c . The results of each iterative cycle are then examined to determine if they are stable or chaotic in behavior. Stable behavior is defined by those points whose values of $|F^n(z)|$ tend to zero or infinity. And chaotic behavior is represented by those points which under iteration of $F(z)$ do not escape to infinity.

To understand how this procedure can be accomplished, consider Listing 1 which contains an algorithm for generating the Julia set, $F(z) = z^2 + c$. The dynamical

process is started by setting z equal to the first point in the (c_r, c_i) complex plane, defined by $(rmin, imin)$ and $(rmax, imax)$. This plane has been subdivided into an $n \times n$ point set which specifies the complex constant c at each point in the resultant image. The absolute magnitude of $|F(z)|$ is then determined and tested to find out if it is stable (i.e., if it escapes to infinity). If not, the current real and imaginary components of $F^n(z)$ are then substituted back into $F(z)$, and the iteration counter is increased by 1. This is done over and over again in an iterative fashion until either the maximum number of iterations is achieved or the condition of $|F^n(z)| \geq 2$ is reached. The associated image pixel gray level is then set to the number of iterations (normalized to 0-255).

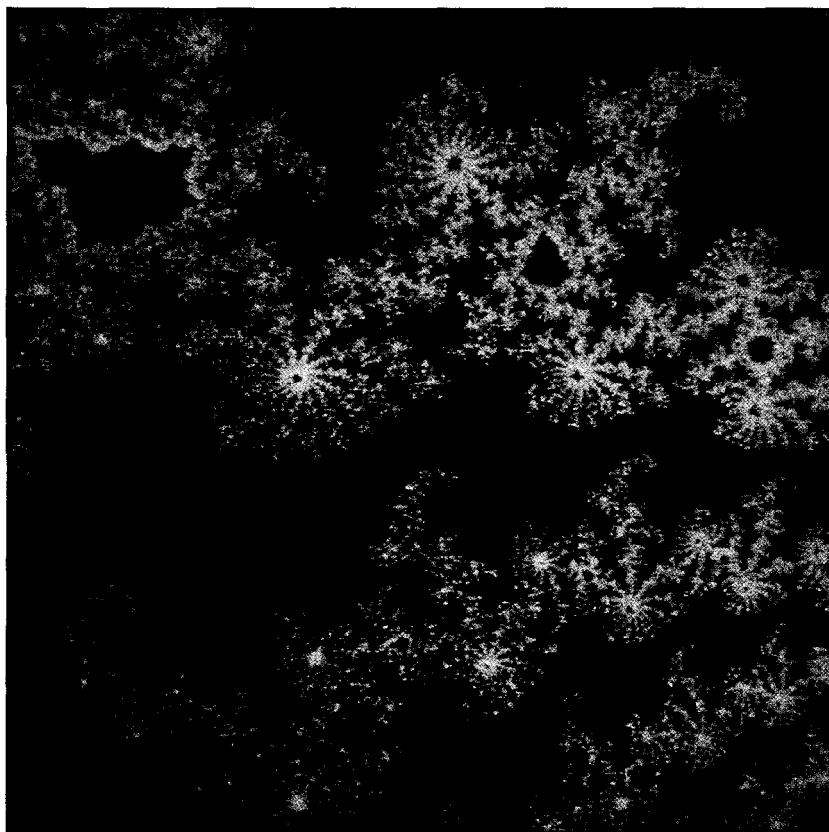
For example, consider the image shown in Photo 1. This image was created with each point in its 512×512 grid ($n = 512$) defined by $|z_r|, |z_i| < 1$ and iterated up to 255 times ($it = 255$). The complex plane defined by c is set to $(rmin, rmax) = (-0.7235, -0.7160)$ and $(imin, imax) = (0.2175, 0.2250)$. Those points where $|z_r|, |z_i| < 1$ and for some number of iterations, it, where

$|F^n(z_r + iz_i)| > 2$, had values that tended to infinity. Such points were therefore not part of the Julia set. On the other hand, points that remained within a circle of radius $|F^n(z)| < 2$, were considered to lie within the so-called Julia set, as well as those points which completed "all" iterations.

FRactal Curves and Plants: Lindenmayer Systems

Within the past four years or so, a very impressive technique called a Lindenmayer system, or L-system for short, has provided a unified approach to creating some really fantastic fractal curves and plants. This L-system procedure is based on the concept of rewriting, where complex objects are created through the successive and recursive replacement of parts of a simpler initial object, according to a set of rewriting rules or productions. Both the initial object and the production rules are defined as character sets that correspond to some graphical representation. In such an algorithm, long character strings composed of letters of the alphabet or special characters such as $+$, $-$, $[$, and $]$ are generated.

They are then interpreted as instructions for the drawing of curves and pictures. The process has had two principle areas of application. These include the generation of fractal curves (e.g., the von Koch snowflake curve and the space-filling curves of Peano and Hilbert), and the realistic modeling of plants. However, the technique has also been used for such exotic applications as designing East Indian art forms or creating graphically motivated algorithms for music composition. It was originally intro-



duced by A. Lindenmayer [3] in 1968 for the purpose of modeling the growth of living organisms, mainly the branching patterns of plants, and it was then applied to the graphical modeling of two- and three-dimensional plants, using a turtle interpretation, by P. Prusinkiewicz in 1986 [4]. It's an impressive procedure that has been widely used and there have been well over a thousand papers written on the subject.

We can gain a quick insight into how the L-system rewriting procedure works by examining the classic example of the Koch "snowflake" curve shown in Figure 1. To create such a fractal curve, we begin by defining two shapes in a simple character notation. The first is the initiator, the so-called "axiom," of the system; the starting point of the procedure. Its character sequence represents the line drawing of the diamond shape shown in Figure 1a. It is defined by the character string "F++F++F++F". The second object is the generator, which in this case corresponds to a single "production rule" (see Figure 1b). It is the object that is used for each rewriting and is defined by the character code "F-F++F-F". Here, F represents a graphically drawn line and each + or - is used to specify a change in the orientation for the drawing of the next line segment. With these two objects, the final complex object is then created by replacing each straight line interval in the axiom with the crooked line defined by the production rule (see Figure 1c). This procedure is then repeated over and over again by substituting the production rule into each line segment of the object derived from

the previous cycle. As a result, the more recursive cycles undertaken, the more complex the object.

Of course not all L-systems are so simple. The contents of the axiom may contain several different commands which require a look-up table of production rules, or there may be a complex set of drawing instructions. However, regardless of the ultimate complexity of each element, the final picture is completely defined by the axiom, the production rules, the number of recursive cycles, and the

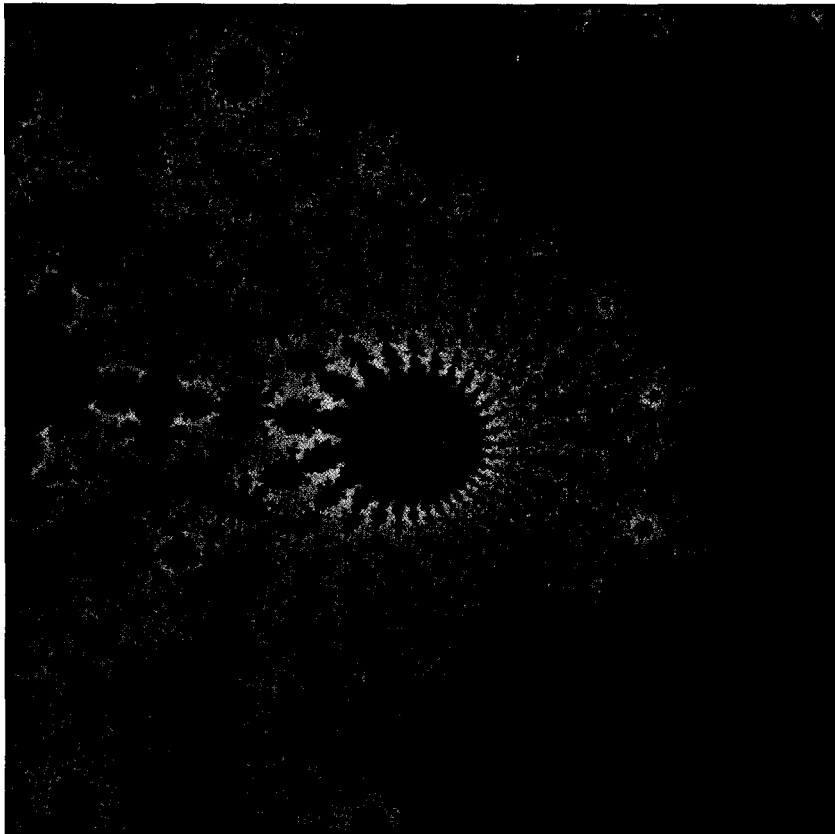


Photo 1 --The Julia set for $F(z) = z^n + c$, with $n = 5$, $it = 255$, $(rmin, rmax) = (-0.7235, -0.7160)$ and $(imin, imax) = (0.2175, 0.2250)$.

definitions of the actions of the "turtle," or drawing pen, which creates the final picture from the instructions within the object's character string. *[Editor's Note: Software for this article is available from the Circuit Cellar BBS and on Software On Disk #17. For downloading and ordering information, see page 708.1]*

Figures 2 and 3 show some examples of what can be created when the procedures I've described are taken one step further. Figure 2 is often referred to as a "pine bush" while Figure 3 shows a generic "bush."

Up to this point we have only considered deterministic fractals, like the Julia sets and Koch curves. But if we are to seriously consider the random irregularities present in such natural phenomena as mountains and rivers, we must apply a stochastic fractal model where the fractal properties apply to the various characteristics of random variables. We will therefore create our terrain model using the stochastic fractal process, fBM, which was introduced by Mandelbrot and Van Ness [5] as a generalization of simple Brownian motion.

Its relevance to our model lies in its ability to produce reasonable statistical properties of real terrain on a first-order approximation. It is not a perfect model, and the results are mixed at best. Terrain definitely displays fractal characteristics, but not over all scales of measurable range. And it is unreasonable for us to expect one simple mathematical model to predict in detail all the results of the many forces and phenomena at work in shaping a planet's

surface, from plate tectonics to rain fall. But for a first approximation, when color and shading are added, Mandelbrot's model, consisting of a sum of randomly distributed faults, is quite impressive. So I propose to discuss some of its basics here, and include a listing of my version of an fBM code (derived from reference [6]) which will enable you to create your own fractal mountain terrains.

There are many algorithms which can be used to compute approximations to fBM. The form that we will

ALGORITHM Julia (rmin, rmax, imin, imax, n, it)

Function:

A fortran program to create Julia set curves, with $F(z)=z^2+c$.

Arguments:

rmin,rmax the low,high real-value of the complex plane **c**
imin,imax the low,high imag-value of the complex plane **c**
n the dimension of fractal image array, nxn
it the number of Iteration cycles

Variables:

rg,ig real,imag Increments of complex constant **c**
cr,ci real,imag parts of the complex constant **c**
zr,zi real,imag parts of **z**
cnt iteration counter
zm absolute magnitude of **z**
zrp,zip real,imag parts of $z^2 + c$

Arrays:

pixel nxn byte image array (0-255 gray scale)

```
byte pixel
dimension pixel(n,n)
real*4 ig
rg= (rmax-rmin)/n
ig= (imax-imin)/n
do 550,i=1,n
do 550,j=1,n
cr=(j*rg)+rmin
ci=(i*ig)+imin
cnt=0
zr=cr
zi=ci
zm=sqrt(zr**2+zi**2)
if(zm.gt.2)then
pixel(i,j)=0
got0 550
endif
do while ((cnt.le.it).and.(zm.lt.2))
zrp=(zr**2 - zi**2 + cr)
zip=(2*zr*zi + ci)
zr=zrp
zi=zip
zm=sqrt(zr**2+zi**2)
cnt=cnt+1
enddo
cnt=256-cnt
if(cnt.gt.127)cnt=-256+cnt
pixel(i,j)=cnt
550:
continue
return
end
```

listing 1 -me Julia set algorithm.

discuss here was introduced by Mandelbrot and Voss in 1985 [7], and is implemented by generating white noise and then filtering it so the results have the frequency distribution characteristic of fBM (or, in the spatial domain, the features of the usual mountain terrain). According to the typical physicist, this white noise can be defined as the unpredictable changes in any quantity Q varying in some time t . And it can be represented by its spectral density, $S_Q(f)$, which provides an estimate of the mean-square fluctuations at a frequency f and, consequently, the variations over a time scale of order $1/f$. The traces made by each of these noises is a frac-

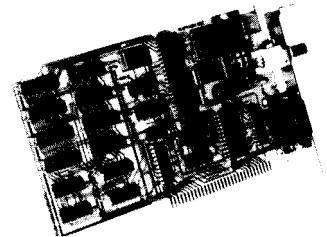
tal curve, where a direct relationship between the fractal dimension and the logarithmic slope of the spectral density exists. fBM curves can therefore be easily constructed from white noise (produced using a pseudorandom number generator) in such a way that they are completely uncorrelated from point to point. The spectral densities of these white noise simulations are flat lines, independent of frequency. But, like "white light," they each contain equal amounts of all frequencies. However, Brownian motion, or random walk as it is often called, is slightly different. It is more correlated, contains more slow (low frequency) than fast (high frequency) fluctuations, and

INTRODUCING On-axisTM 4 CHANNEL 24-bit QUADRATURE-MODE COUNTER

Acquires and displays position information from optical encoders. Resolution is four times the encoder. Complete with demo software and driver source code.

\$299⁰⁰ INTRODUCTORY PRICE
(add \$150 for optional connector to Bauch & Lomb "glass scales".)

étudeTM 25 MHz 8-bit ANALOG-TO-DIGITAL CONVERTER



Based on the TRW THC1068 hybrid flash converter, its high signal-to-noise ratio yields excellent accuracy at the Nyquist limit.

- 4 KB of cache SRAM or host as converted at DMA speed
- I/O or DMA data transfer
- 10 MHz full-power bandwidth
- 3.92 mV resolution
- Factory calibrated
- 16 jumper selectable base addresses
- External clock and trigger inputs
- TTL compatible
- Software source code included

PRICE: **\$495⁰⁰**

Each product requires: PC compatible 1/2 length 8-bit expansion slot, DOS 2.11 or greater, EGA, VGA or Hercules display needed for graphic representation of data

Silicon Alley Inc.

**P. O. BOX 59593
RENTON, WA 98058
(206) 255-7410**

©1990 Silicon Alley Inc. étude and on-axis are trademarks of Silicon Alley Inc. Other brand or product names are trademarks or registered trademarks of their respective holders. Prices & specifications subject to change.

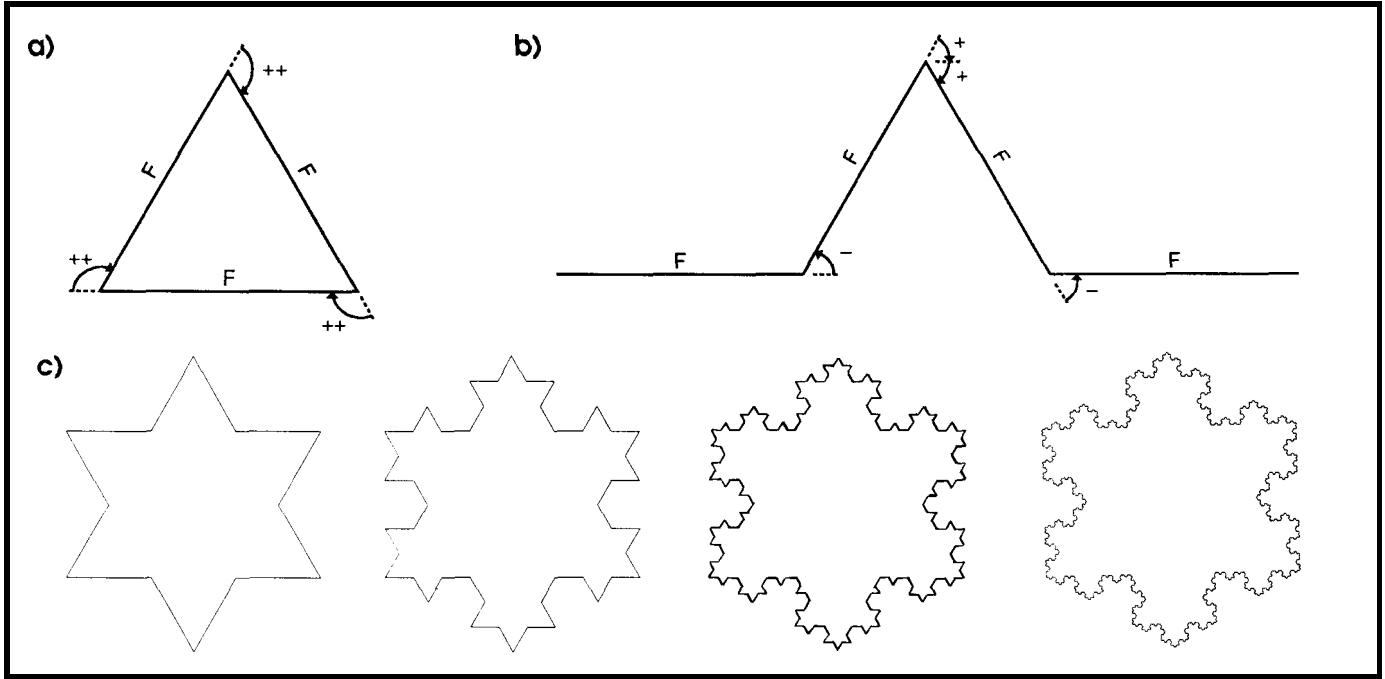


Figure 1 — The L-system for the von Koch snowflake curve is defined by the axiom (a) "F++F++F++F" and the production rule (b) where "F" goes to "F-F++F-F". The first, second, third, and fifth stages in the curve generation are shown in (c).

has a steep spectral density which varies as $1/f^B$. As such, it can be formally thought of as the integral of the white noise. With this in mind, the implementation of our fBM model [6] is easily achieved by performing a simple numerical integration of computer-generated white Gaussian noise in a two-dimensional format. This is accomplished by using the Fourier filtering method [6] for a modeled spectral density S which depends on the two frequency variables u and v corresponding to the x and y directions. Here, we assume that all directions within the modeled x - y plane are equivalent with respect to their statistical properties, and that its spectral density, S , depends only on $(u^2 + v^2)^{0.5}$. Where, if this surface is then sampled along any straight line in the x - y plane, the spectral density S of this fBM (in this one-dimensional representation) goes as a power law of $1/f^B$, with $0.5 < B < 1.5$. This spectral density therefore behaves like,

$$S(u,v) = 1 / (u^2 + v^2)^{H+1} \quad (3)$$

where the parameter H is used to determine the fractal dimension,

$$D = 3 - H, \quad 0 < H < 1 \quad (4)$$

and the two-dimensional discrete inverse Fourier transform of $S(u,v)$ defined as,

$$fdata(x,y) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} data_{ij} e^{2\pi i(i x + j y)} \quad (5)$$

for $x, y = 0, 1/N, 2/N, \dots, (N-1)/N$. Thus, the expected value of the coefficients $data_{ij}$ are approximately,

$$E(|data_{ij}|^2) \propto ((i^2 + j^2)^{H+1})^{-1} \quad (6)$$

Note, since the constructed function $fdata$ is real, it must satisfy the necessary conjugate symmetry conditions.

Listing 2 is an example of the computer implementation of the above-described spectral synthesis fractional Brownian motion technique. It required the generation of random variables with a normal, or Gaussian, distribution having a mean of zero and a variance of one. I made use of my system random number generator to provide me with random

numbers that were uniformly distributed over the interval $[0, A]$, for $A = (2^{31} - 1)$, with my Gaussian random variables being generated by taking linearly scaled averages of these system random numbers and then standardizing them by subtracting their respective expected values and dividing by their standard deviations. The polar coordinates of the i^{th}, j^{th} Fourier coefficient were then determined ($rad, phase$) and the components of a vector from the origin to that point were stored in the i^{th}, j^{th} position of the spectral density array ($data(i,j)$). This 2-D spectral density array was then

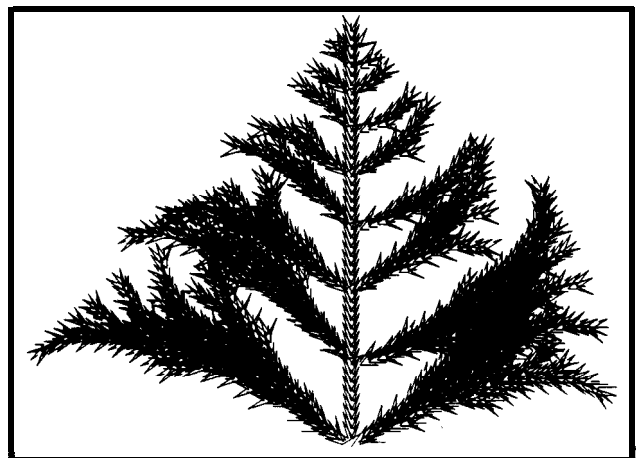


Figure 2—A pine bush (maxlev = 11) with axiom "SLFFF", angle delta = 18°, and using the production rules. "S" goes to "(+++G)(-G)TS", "G" goes to "+H(-G)L", "H" goes to "-G(+H)L", "T" goes to "TL", and "L" goes to "(-FFF)(+FFF)F".

mapped back into a spatial domain (corresponding to x,y positions) representation through the use of a 2-D inverse fast Fourier transform. This inverse FFT was accomplished by first taking the line FFT of each row within $data(i,j)$, transposing the results by exchanging **rows** and **columns**, and then taking the line FFT of each row again. The power spectrum, or log amplitude, of this complex array was then determined and renormalized on a 0-255 gray scale and then stored in the real array $fdata$. This much-sought-after fractal mountain terrain was then sent to a system-dependent 3-D plot routine (not included here) for display.

Figure 4 demonstrates the fractal-dimensional dependence, defined by the parameter H , for $0 < H < 1$, in a sequence of fBM terrains generated using the same random number seed, starting from $H = 0.4, 0.5, 0.7, \text{ to } 0.9$. The difference in their associated "detail" can be attributed to their "space filling nature" defined by the fractal dimension. Since $D = 3 - H$, it can be seen that D equals 2.6, 2.5, 2.3, and 2.1, respectively, with the expected detail of each spectral synthesis diminishing as H increases and D decreases.

FRACTAL PLANETOIDS

I would like to show you how you can use the above fBM output to create your own fantastic planet pictures using a version of the planet-generating algorithm (available on the BBS). This algorithm is designed to use many of the image synthesis techniques discussed in the article "Image Synthesis: A Tutorial" which appeared in *CIRCUIT CELLAR INK # 12*. These rendering procedures include such forms as proper viewing perspective, three-dimensional image projection onto a two-dimensional **display plane**, source lighting effects, surface shading, hidden surface removal, colorization

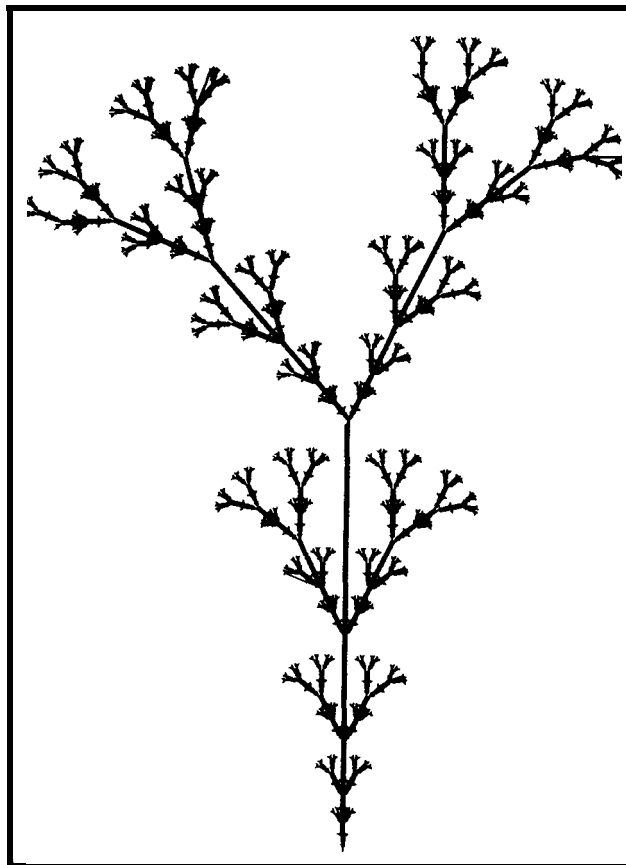


Figure 3-A bush (maxlev = 8) with axiom "G", angle delta = 25.7°, and the production rules, "G" goes to "GFX(+G)(-G)" and "X" goes to "X(-FFF)(+FFF)FX".

banding, and simple antialiasing, which are easily implemented using both a reference block and a look-up table format.

To begin the creation of a fractal world, a model of the continental land masses derived from the fBM code is needed. Since land masses have very little structure when seen from outer-space, it is recommended that a high fractal dimension not be used. We do not want a space-filling, jagged mountain range or rugged plains. Instead, using a value of H which ranges between 0.8 and 1.5 provides the best results (we can trick the fBM code into giving exceptionally smooth shapes by letting H be greater than 1.0). Here, the fBM image will contain enough structure to be reasonably similar to mountain ranges, while at the same time have sufficient smoothness to resemble the structure of coastal plateaus and interior plains. To create oceans, a constant threshold level is subtracted from the fBM image to create large areas of zero value be-

tween the desired land masses. The image is then renormalized on a scale of zero to approximately 5% of the base planetoid radius, $(0 - 0.05 \times r_{sph})$, to enable proper scaling when the fBM surface structure is added to the surface of the base planetoid sphere.

One example of a fBM continental land mass image is shown in Photo 2a. Here, H was chosen to be 1.1, a threshold of 180 was subtracted from the 0-255 level original, and it was then normalized on a scale of 0 to 6. An example of a fractal world created from this image is shown in Photo 2b. The model of the continental land masses in 2a was stereographically projected onto a 3-D sphere using a form of Riemann mapping. Each point on the planet's surface which was visible to the viewer, as determined by his or her perspective, was then projected onto a 2-D image display plane and evaluated

for surface shading, hidden surface removal, sunlighting effects, and colonization coding; in this case, a black-and-white 0-255 continuous gray scale. To see how this image was created, I suggest we now review how each rendering technique was implemented within the planet code.

BASIC GEOMETRICAL CONFIGURATION

The fractal planet generator algorithm PLANET is based on a three-dimensional vector ray-trace system. It is made up of four individual components: the light source, a viewpoint eye, the planetoid, and a display screen, with each component having its own 3-D orthogonal coordinate system related to each other through matrix transformations. The primary axis of the system is along the eye-to-display vector and is defined in terms of the sphere's coordinate system. Its orientation is specified by the input characteristics of eye point-of-view

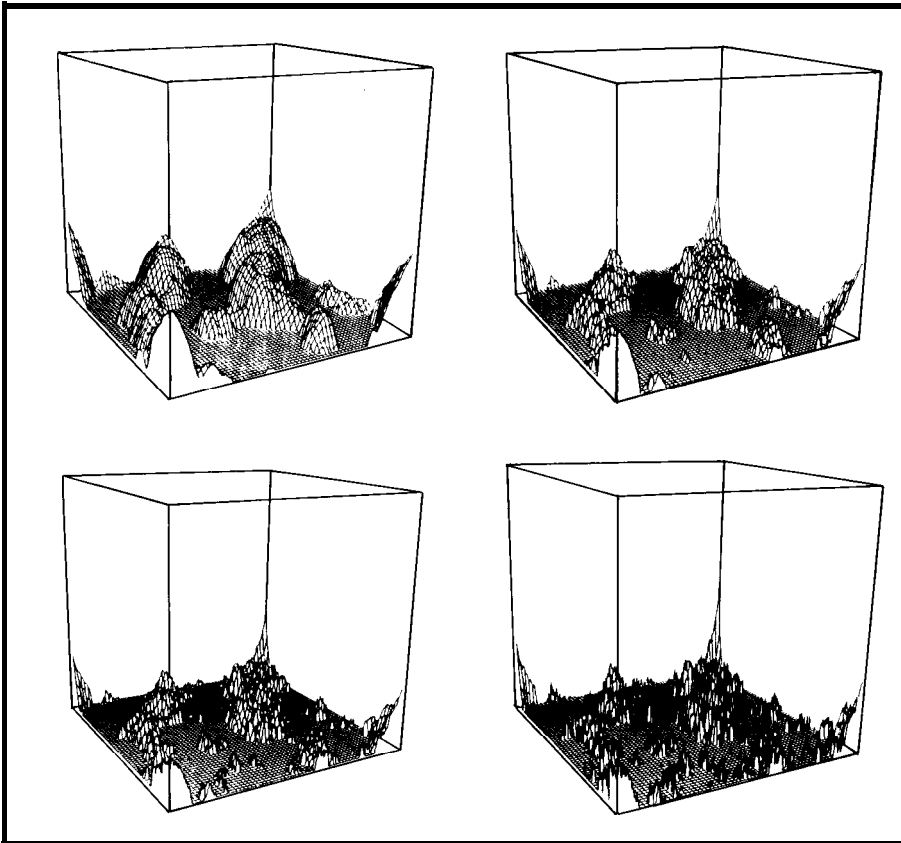


Figure 4—The spectral synthesis of a mountain range with the fractal dimension varying from 2.6 to 2.1.

spherical vector coordinates $(\text{reye}, \theta, \phi)$, where reye is the radius vector from the center of the planet sphere to the eye, θ is the tilt angle of reye from the $+z$ axis (Z_s), and the angle ϕ represents a rotation of reye about the z axis in the x - y plane ($X_s - Y_s$). The source location is defined by the spherical vector coordinate $(\text{reye}, \theta_s, \phi_s)$. Its magnitude is chosen to correspond to the eye vector for convenience only, but its direction is uniquely defined by the tilt and rotation angles, θ_s and ϕ_s . The origin of the display screen coordinate system is defined by a vector dis whose orientation in space is along the vector reye . It has spherical coordinates of $(\text{dis}, \theta, \phi)$ as defined in the planet's system. If dis is less than reye , then the resultant 2-D projection is reduced. If dis is greater than reye then the 2-D projection of the planet is magnified. The coordinates of the projection points on the display screen are defined by the components $(X_d, Y_d, 0)$, where the z axis is along the dis vector. This display screen can also be tilted about its z axis, Z_d , or

Desktop 9-Track Tape Subsystem

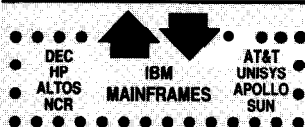
Now, **9-track tape** lets your micro exchange data with minis and mainframes



Simply exchange data files on a reel of 9-track tape.

9-TRACK is the first choice for file interchange among data processing professionals. Now, Qualstar's low cost 1/2-inch 9-track Streaming tape systems bring full ANSI data interchange to IBMPCs or Macintosh, giving your micro the freedom to exchange data files with nearly any mainframe or minicomputer in the world.

Available in both 7" and 10-1/2" versions, compact Qualstar tape drives can sit on your desktop, using less space than an ordinary sheet of paper. Systems include DOS or XENIX compatible software copier card and cables. High reliability 1600 or 6256 BPI capability may be used for disk backup as well as data interchange. Discover the big advantage 9-track tape has over other micro/mainframe links.



#1 Selling 9-Track Systems on the Desktop



QUALSTAR

9621 Irondale Ave., Chatsworth, CA 91311

©1989 Qualstar Corp. All product and company names and trademarks are the exclusive property of their respective owners.

Call us today!
FOR DETAILS AND TO ORDER
FAX (818) 882-4881
PHONE (818) 882-5822

Mandelbrot Explorer 3.3

Order some Chaos!

Fantastic 32-color fractal graphics on 16-color EGA/VGA cards to 800x600... 16.5 trillion power magnification... easy stop, start, save and retrieve... montages... dwell count files... stereo pairs... auto color gradients... zoom box... real-time display of periodic orbits... auto-backup... PCX image file support... all optimizations for speed including pixel interpolation and both 16- and 32-bit integer support includes 7 ready-made images for immediate gratification.

\$35.00

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

Please specify disk format
Overseas orders please add \$4

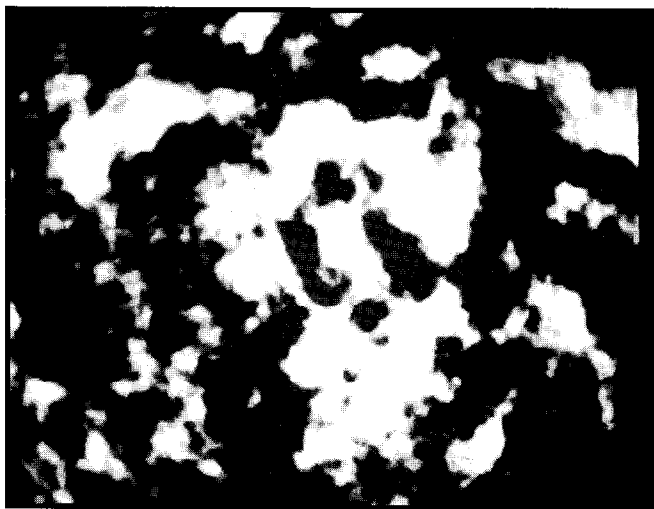


Photo 2a—A simple contour image of the 2-D terrain map generated by the fBM algorithm using a fictitious $H = 1.1$.

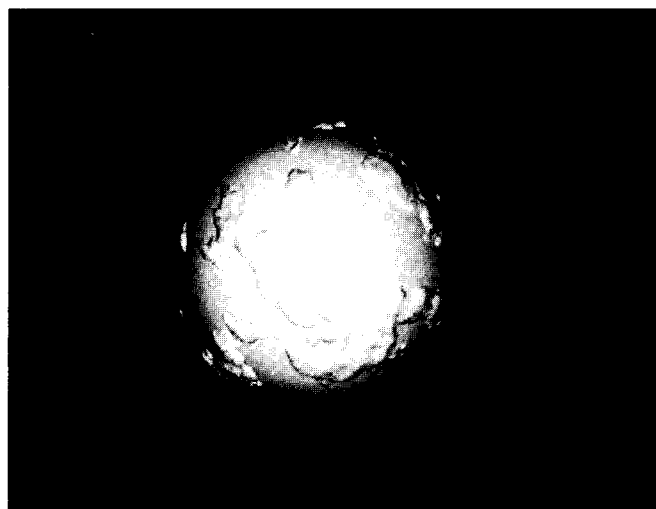


Photo 2b—The fractal planet generated using the contour shown in (a).

rotated about the y axis, Y_d , in the X_d, Z_d plane by specifying the input angles α and β .

THE 3-D FRACTAL PLANET

Each 3-D fractal planet is created by wrapping the fBM image map around the reference sphere using a modified Riemann mapping technique. This is accomplished with the help of a stereographic projection between the fBM image plane and the planetoid reference sphere in the following fashion: A sphere of radius $rsph$ is constructed such that the fBM image plane is tangential to it at the origin of the fBM plane, as shown in Figure 5. The point T on the top of the sphere opposite the fBM origin is used as the "eye" of the stereographic projection. Lines from T are then drawn to intersect both the sphere (at point I) and the fBM plane at each pixel point $(X_p, Y_p) = P$ in the fBM plane. A simple mapping of point P onto the

sphere intersection point can then be achieved by adding the value of the fBM image pixel at point P (normalized to a 0-6 scale for $rsph = 128$) to the magnitude of the radius of the base sphere $rsph$ at the intersection point I . The new coordinates of this mapped surface are then specified by (X_s, Y_s, Z_s) whose radius is $rsph + fBM(X_p, Y_p)$. X_s, Y_s , and Z_s are defined by the

sphere's coordinate system whose origin is at the center.

Figure 6 demonstrates how the point (X_s, Y_s, Z_s) is determined in PLANET's subroutine `PRO J`. The procedure involves the determination of the direction angles O_s and P_2 which define a vector drawn from the origin of the planetoid sphere to the intersection point I . This is accomplished by

constructing a reference right triangle using the projection of the line drawn from T to the point P onto the $x-y$ plane. The angles O_2 and O_s are then easily determined and the angle P_2 is found by evaluating the rotation of this reference triangle in the $x-y$ plane of the fBM image file.

The radius of a point on the surface is defined as,

$$RD = rsph + fBM(X_p, Y_p) \quad (7)$$

which is a summation of the radius of the sphere and the normalized value of the fBM terrain map at point (X_p, Y_p) . Since the spherical coordinate angles O_s



Photo 3—An fBM image mapped onto a sphere and shaded according to Eqn. 16. The position of the source was chosen 35 degrees offset from the observer and the illumination values were banded and normalized over a range of 0-255 with the seas (for $r = rsph$) ranging from 1 to 127 and the land masses from 128 to 255.



Photo 4-A fractal planet and moon. The planet was set up for color banding and created using $reye = 3027$, $dis = 3027$ (magnification = 1), $q = 180$, $f = 0$, $qs = 155$, $fs = 0$, $rsph = 128$, $lnt = 1$, and $/color = 1$. The moon was created by changing the fBM file, using a continuous gray scale ($lcolor = 0$), and changing dis to 1024 (magnification = 0.34), thereby reducing the moon size.

and $P2$ are the same for this (radially extended) surface point, the Cartesian coordinates for it can be easily determined using:

$$\begin{aligned} X_s &= RD \sin(O_s) \cos(P2) \\ Y_s &= RD \sin(O_s) \sin(P2) \\ Z_s &= RD \cos(O_s) \end{aligned} \quad (8)$$

This surface point can now be evaluated with respect to the point-of-view of the observer and the light source, then projected onto the display image plane.

2-D IMAGE PROJECTION

The display image is created in a step-by-step process during the surface mapping sequence. This is done to minimize memory space allocation since creating a 3-D object would require an $N \times N \times N$ array. However, each surface point is evaluated only once, yet it is necessary to use the

properties of each point several times. As a result, a look-up table ($surf$) was created to enable dynamic storage of each visible surface point. The actual display image projection sequence applied in PLANET uses the following steps: For an fBM image point (X_p, Y_p), the planetoid mapping point (X_s, Y_s, Z_s) is determined by $PRO \sigma$ and the components of a normal vector from the origin of the sphere through this surface point is defined as,

$$\begin{aligned} X_{sn} &= X_s / R_s \\ Y_{sn} &= Y_s / R_s \\ Z_{sn} &= Z_s / R_s \end{aligned} \quad (9)$$

The components of the visibility vector from the surface point to the eye position defined by the vector $reye$ are then determined as,

$$\begin{aligned} V_x &= R_x - X_s \\ V_y &= R_y - Y_s \\ V_z &= R_z - Z_s \end{aligned} \quad (10)$$

And a determination of whether the surface is in view is made by finding the dot product between the normal vector and the vision vector V ,

$$\begin{aligned} V \bullet N &= V_x X_{sn} \\ &+ V_y Y_{sn} \\ &+ V_z Z_{sn} \end{aligned} \quad (11)$$

If the dot product is greater than zero, then the surface is assumed to be visible to the observer. The surface point is then converted into the observer's system using the following rotation-tilt matrix transformation,

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \sin(\phi) & -\cos(\phi) & 0 \\ -\cos(\phi)\cos(\theta) & -\sin(\phi)\cos(\theta) & \sin(\theta) \\ -\cos(\phi)\sin(\theta) & -\sin(\phi)\sin(\theta) & -\cos(\theta) \end{bmatrix} \begin{bmatrix} X_s \\ Y_s \\ Z_s \end{bmatrix} \quad (12)$$

and a vector is then drawn from the eye through the transformed surface point (X, Y, Z) to intersect with the display plane at point (X_d, Y_d).

The equation for the display plane is determined by using the basic properties of the cross product. Three points are defined in the display plane, one at the origin and the other two along the X_d and Y_d axis, respectively. They are defined by the points $(0, 0, dis)$, $(0, 10, dis)$, and $(10, 0, dis)$ in such a way that a vector joining any pair of these points lies in the display plane and the cross product (CV_x, CV_y, CV_z) of any two such defined vectors is perpendicular to that plane. The equation of the display plane can then be defined as,

$$\begin{aligned} cv_x \times (X - 10) \\ + cv_y \times (Y) \\ + CV_z \times (Z - dis) = 0 \end{aligned} \quad (13)$$

with normal direction vector CV through the point $(10, 0, dis)$. The x-y coordinates of the projection interception point can now be determined from the following equations:

$$\begin{aligned} V_d &= (X^2 + Y^2 + Z^2)^{0.5} \\ B &= CV_x \times (X/V_d) \\ &+ CV_y \times (Y/V_d) \\ &+ CV_z \times (Z/V_d) \\ R &= (10 \times CV_x + dis \times CV_z) / B \\ X_d &= -R \times X / V_d \\ Y_d &= -R \times Y / V_d \end{aligned} \quad (14)$$

ALGORITHM fBM

function: to model terrain based on the stochastic fractal process called fractional **Brownian Motion (fBM)** using a Fourier (spectral synthesis) filtering technique.

Globalsarand range of random number set $[0,A]$, wt A typically $2^{31}-1$ or 2^3-1
 seed seed value for random # generator $\gg 1$, odd integer
 nrand number of random samples to be taken
 gadd real parameter for linear transformation in gaussian random number generation
 gfac real parameter for linear transformation in gaussian random number generation
 rnd result of system random number function
 gaus0 gaussian random number
 rad,phase polar coordinates of Fourier coefficient
 n size of doubly indexed array of complex variables
 H parameter $0 < H < 1$, determines fractal dimension $D=3-2H$

Arrays data complex 512x512 array containing the spectral synthesized terrain
 fdata the log amplitude of the inverse Fourier transform normalized to 0-to-255 gray levels for plotting

PROGRAM: fBM fractional Brownian Motion Terrain Modeling

```

program fBM
integer*4 seed
common arand,nrand,gadd,gfac,seed,sum,gaus0,rnd
real*4 fdata(512,512)
complex data(512,512),cmplx
intrinsic cmplx

write(6,5)
5 format(1x,'Input H, seed, where 0<H<1:','$)
read(5,*)h,seed ! input random # seed, fractal dim
nrand = 4 ! set number of random samples
arand = 2147483647.0 ! set size A=231-1
  
```

(continued)

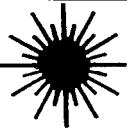
listing 2-Terrain modeling using fractional Brownian Motion.

The coordinates (Xd,Yd) are then converted into array indices and the gray value of the map-projected surface point is entered into the display D IS P array.

SHADING, LIGHTING, AND HIDDEN-SURFACE EFFECTS

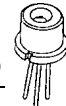
The actual value entered into the display image array depends on the evaluation of the corresponding mapped surface point as to its shading and sourcelight illumination. After each mapped point passes the visibility test in Equation 11, its image gray level is determined. This is accomplished by studying the shape of the surface of the planet in the proximity of that point and then determining the amount of light reaching it from the source. Two additional points adjacent to the surface point (Xs,Ys,Zs) are mapped onto the sphere, from (Xp+1,Yp) and (Xp,Yp+1) in the fBM image, and two vectors on surface of the sphere defined by these three points are defined. A unit normal

LASERS



NEW VISIBLE LASER DIODES

Toshiba TOLD 9200.
 3 mw, 670 nm Red output.
 Cat. #LDV 9200 \$75.00



NEW POWER SUPPLY

for Toshiba — 9200 series diodes,
 4.5V input, .75" x .6" dia.
 Cat. # LDD-92 \$35.00

NEW COLLIMATING LENS

for visible diodes. Just
 press over diode.
 Cat. #LDC-6 \$20.00



NEW VISIBLE DIODE MODULE

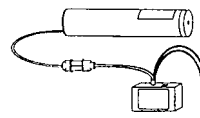
0.8 mW typical
 output power
 @ 670nm (red)

Factory made module that
 includes the diode, collimator,
 and power supply, all in a 1.6" x
 .63" diameter housing. Operates
 on 3.9 to 5 VDC @ 80mA.
 Cat. # LDM-001 \$150.00



HELIUM NEON LASER KITS

Kits consist of matching
 laser head and power
 supply, along with a
 user's manual. All kits
 are FDA approved.



.95mW, 633nm (Red) output with 12VDC input
 power supply. Draws .85 amps.
 Cat. #HNKD-95 \$110.00

.95mW Kit with 110VAC input power supply.
 Cat. #HNKA-95 \$130.00

4.5mW, 633nm (Red) output with 12VDC input
 power supply. Draws 1.5 amps.
 Cat. #HNKD-50 \$200.00

4.5mW Kit with 110VAC input power supply.
 Cat. #HNKA-50 \$220.00

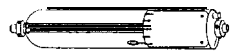
NEW 6.5mW 633 (Red) output with 12VDC input
 power supply. Draws 1.5 amps.
 Cat. #HNKD-700 \$230.00

6.5mW Kit with 110VAC input power supply.
 Cat. #HNKA-100 \$250.00

NEW 50mW 543 (Green) output with 110VAC
 input power supply.
 Cat. #6HNKA-10 \$465.00

HELIUM-NEON LASER TUBES & HEADS

New hard sealed units



.5 - 1.0 mW TUBE. Operates on 1300VDC @
 4.0mA. Dimensions: 5.8" x 1.0" diameter.
 Cat. #06 \$35.00

5-7mW TUBE. Operates on 2200VDC @ 6.5mA.
 Dimensions: 13.8" x 1.45" diameter.
 Cat. #50 \$100.00

NEW 2 - 3 mW POLARIZED HEAD

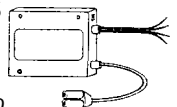
Operates on 1800VDC @ 6.5mA.
 Dimensions: 10.8" x 1.75" diameter.
 Cat. #230HP \$90.00

HELIUM NEON POWER SUPPLIES

New, factory made, switchers. 10-14VDC input
 micro P/S for 06 tube. Only 2.1" x .88" dia.
 Cat. #12-B \$75.00

ADJUSTABLE SUPPLIES

From 1.2 to 2.6KV, 4-6.5mA.
 For .5 to 7mW HeNe Lasers.
 12VDC input supply.
 Cat. #12-C \$75.00
 110VAC input supply Cat. #110ADJ \$95.00



Call or write today to receive a **FREE CATALOG** which includes
 Optics, Holography, Kits, Scanners, Books, and more.

Meredith Instruments

"THE SOURCE FOR LASER SURPLUS"

6403 N. 59th Avenue / Glendale, Arizona 85301 • (602) 934-9387 • P.O. Box 1724 | Glendale, Arizona 85311

passing through (Xs,Ys,Zs) to this surface segment is then determined.

Since the source can be positioned independently from the observer, a new source illumination vector (Vxs,Vys,Vzs) is determined,

$$\begin{aligned} Vxs &= Rxs - Xs \\ Vys &= Rys - Ys \\ Vzs &= Rzs - Zs \end{aligned} \quad (15)$$

where (Rxs,Rys,Rzs) are the components of the light source specified by the spherical coordinates (reye,θs,φs). The angle of incidence between the direction of the illumination vector (Vxs,Vys,Vzs) and the surface normal is then determined by using the properties of the dot product. This angle is important since according to Lambert's cosine law, the intensity of the reflected light is proportional to its cosine. And since we also expect the brightness of an illuminated surface to depend on the distance from the light source, the PLANET algorithm uses the following simple lighting-shading model:

$$Sint = 10 \times \cos O / (Vs + 0.001) \quad (16)$$

where *Sint* is the intensity of illumination of the point (Xs,Ys,Zs) on the planet's surface, *I₀* is the initial intensity of the light source at (reye,θs,φs), *cosO* is the cosine of the angle between the surface segment normal and a vector drawn from the source to the point (Xs,Ys,Zs), and *Vs* is the distance of the light source from the illuminated point (see Photo 3).

Of course the next question is whether or not this illuminated point is truly viewed by the observer, or is it obscured by some other surface feature (e.g., a mountain peak). This is the usual hidden-surface problem and it is simply solved through the use of a look-up table (SURF) and a reference array block (SURFP). Whenever the relative illumination (gray level) of a mapped point is evaluated, all important information about that point is stored in the look-up table. This table is number indexed by the visibility counter *iv*, which is incremented whenever a surface point is determined as visible to the observer. This

```

gadd = 3.46410          ! set gadd = (3*nrnd) .5
gfac = 2*gadd/(nrnd*arand) ! set linear transform factor
do i=0, n-1
  do j=0, n-1
    if(i.eq.0 .and. j.eq.0)then
      rad=0.0
    else
      ! find gaussian rand num with norm
      ! distr (mean 0 and variance 1)
      sum=0
      ! set summation = 0
      do k=1,nrand
        ! random sample nrnd times
        rnd=ran(seed)
        ! system call for random number
        sum=sum+rnd*arand ! sum samples
      enddo
      gaus0 = gfac*sum-gadd ! det integ gaus rand number
      ! polar coords of Fourier Coeffs
      rad = ((i*i+j*j)**(-.5*(h+1)))*gaus0
      phase = 2*3.141592*rnd
      data(i,j)=cmplx(rad*cos(phase), rad*sin(phase))
    endif
  enddo
enddo
! end creation of coefficients
call invfft(n,data)
! find inverse Fourier transform
do j=1, n
! find pwr spect of modeled terrain
do i=1, n
  x = real(data(i,j))
  y = aimag(data(i,j))
  xf = sqrt(x**2 + y**2)
  ! amplitude of complex point
  if(xf.eq.0)xf=1
  fdata(i,j)=alog10(xf)
  ! log of amplitude
  if(i.eq.1 .and. j.eq.1)then! test for first point
    fhi=fdata(i,j)
    flo=fdata(i,j)
  endif
  fhi=max(fdata(i,j),fhi)
  ! find maximum value
  flo=min(fdata(i,j),flo)
  ! find minimum
enddo
enddo
biti=(fhi-flo)/255
! set renormalization constant
do j=1,n
! renormalize data
do i=1,n
  fdata(i,j)=int((fdata(i,j)-flo)/biti)
enddo
enddo
call plotter(fdata)
! send synth terrain to plotter
! device dependent so not included

end

SUBROUTINE: find inverse P-Dimensional FFT
function: to find the inverse 2-D FFT of an input image/function
stored in the 2-D complex array "data(i,j)" with the
results of the operation being stored back into the
original complex array and the initial data lost.

The factor (-1)**(i+j) is used to insure the proper
relationship between the components of the FFT. If it
is not used, the resultant FFT will seem to be segmented
into 4 displaced sections about the origin.

Arrays  xr  the real part of a row within the input complex
         xi  the imaginary part of the row used in xr, within
         data a 512x512 complex array containing the spectral
         bufr a temporary transfer buffer

subroutine invfft(n,data)
dimension xr(512),xi(512)
complex data(512,512),bufr(512,512),cmplx
intrinsic cmplx
do 50,j=1,n
! first fft and transpose
do i=1,n
  xr(i)=real(data(i,j))*(-1)**(i+j)
  xi(i)=aimag(data(i,j))*(-1)**(i+j)
enddo
call fft1(n,xr,xi)
do 50,i=1,n
  bufr(j,i)=cmplx(xr(i),xi(i))
50 continue
do 100,j=1,n
! second fft
do i=1,n

```

(continued)

listing 2-continued

```

      xi(i)=aimag(bufrr(i,j))
    enddo
    call fft1(n,xr,xi)
    do 100,i=1,n
      data(i,j)=cmplx(xr(i)/n**2,xi(i)/n**2)
100 continue
    return
  end

```

SUBROUTINE: find line FFT

function: to find the fast Fourier transform of a line of data, from E.Oran Brigham's book "The Fast Fourier transform" Prentice-Hall, Inc, Englewood Cliffs, NJ, 1974, page 164. The input into the FFTL program are: XR, the real part of the function to be discrete Fourier transformed; XI, the imaginary part; and N, the number of points; with NU being defined as N=2**NU. Upon completion, XR is the real part of the transform and XI is the Imaginary part of the transform. Input data is destroyed. For more information I refer you to Brigham's book

```

subroutine fft1(n,xr,xi)
dimension xr(512),xi(512)
nu=9
n2=n/2
n1=nu-1
k=0
do 100 l=1,nu
02 do 101 i=1,n2
  np=nu
  p=ibitr(k/2**n1,np)
  arg=6.283185*p/float(n)
  c=cos(arg)
  s=sin(arg)
  k1=k+1

```

```

      k1n2=k1+n2
      treal=xr(k1n2)*c+xi(k1n2)*s
      timag=xi(k1n2)*c-xr(k1n2)*s
      xr(k1n2)=xr(k1)-treal
      xi(k1n2)=xi(k1)-timag
      xr(k1)=xr(k1)+treal
      xi(k1)=xi(k1)+timag
01 k=k+1
  k=k+n2
  if(k.lt.n) go to 102
  k=0
  n1=n1-1
00 n2=n2/2
  do 103 k=1,n
  np=nu
  i=ibitr(k-1,np)+1
  if(i.le.k)goto 103
  treal=xr(k)
  timag=xi(k)
  xr(k)=xr(i)
  xi(k)=xi(i)
  xr(i)=treal
  xi(i)=timag
03 continue
  return
end
function ibitr(j,np)
j1=j
ibitr=0
do 200 i=1,np
j2=j1/2
ibitr=ibitr*2+(j1-2*j2)
00 j1=j2
return
end

```

listing 2-continued

68000 K-System Factory Direct Prices

Now there is a bus that makes it easy to use the entire family of 68000 components. Utilizing native 68000 signals, the K-Bus makes it possible to create low cost 68000 systems in a straightforward manner. The simplicity inherent in the K-System concept allows the system designer the ability to concentrate on meeting the demands of the applications. This same simplicity combined with its low cost makes the K-System ideal for applications ranging from personal use through educational and laboratory applications up to industrial control and systems development. All of this is accomplished at no sacrifice in performance or reliability.

The convenient size (4 x 5 1/4 inch) of the K-Bus boards permits the optimal division of system functions thus simplifying system configuration. The motherboard incorporates integral card guides and compatible power connectors which minimizes packaging requirements. Both SKDOS and OS-9/68000 are fully supported allowing efficient system utilization in both single and multi-user applications.

AVAILABLE IN KIT FORM

Boards currently in production:

K-BUS	12 Slots, .8" centers, PC type power connectors	\$129.95
K-CPU-68K	10MHz 68000 CPU, 2 ROM sockets (12 or 16MHz)	\$129.95
K-MEM	256K static RAM or 27256 type EPROMs (OK installed)	\$59.95
K-ACI	2 serial ports with full modem control (68681)	\$99.95
K-FDC	Floppy disk controller (up to four 5 1/4 drives)	\$99.95
K-SCSI	Full SCSI implementation using 5380 chip	\$99.96
K-DMA	2 channel DMA controller using 68440 chip	\$129.96
K-PROTO	General purpose/wirewrap board	\$39.95
K-xxx-BE	Bare board with documentation lorabov	\$39.95

Software:

SKDOS	Single user, editor, assembler, utilities, BASIC	\$150.00
OS-9/68000	Multi-user, editor, assembler, SCRED, utilities BASIC, C, PASCAL, FORTRAN are available	\$300.00

Inquire about our UniQuad line of 68xxx Single Board Computers.

Quantity and package discounts available

Terms: Check, Money Order, Visa, MasterCard—Prices include UPS ground shipment in continental US.

Hazelwood Computer Systems

Highway 94 at Bluffton UniQuad™ K-Kits™
Rhineland, MO 65069 • (314) 236-4372

Order Service #138

BTK52 BASIC-52 TOOLKIT

The BTK52 is an intelligent front end for program development on the MCS BASIC-52 CPU. It reduces 8052 program development time substantially and can be used with any MCS BASIC-52 based target system. The BTK52 runs on any IBM-PC/XT or compatible.

- Program download from PC host to target
- Program upload from target to PC host
- BASK program renumber utility, with "from," "through," "start," and "increment"
- Full screen program editing
- Single line editing with automatic error line number detection
- Full on-line help facility
- Transparent, adaptive line compression for full input line buffer utilization
- All functions accessible with only one keystroke from the terminal emulator
- \$125

BXC51 8051/8052 BASIC COMPILER

- Fully compatible with code written for MCS BASIC-52 interpreter
- Now with integer, byte and bit extensions for code that runs more than 50 times faster than the MSC BASIC-52 interpreter
- Full floating point support
- In-line assembly language option
- Compile time switch to select 8051/8031 or 8052/8032 CPUs
- Includes Binary Technology's SXA-5 1 cross-assembler and Hex file manipulation utility
- Compatible with any RAM or ROM memory mapping
- Runs on IBM-PC/XT or compatible
- \$295

603-469-3232 • FAX 603-469-3530



Binary Technology, Inc.

Main Street • PO Box 67 • Meriden, NH 03770



Order Service #112

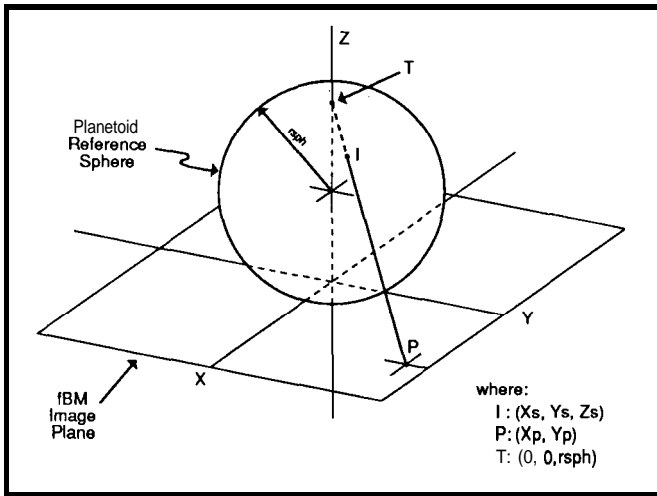


Figure 5—The stereographic projection between the fBM image plane and the planetoid reference sphere of radius $rsph$ is constructed such that the fBM image plane is tangential to the sphere at the origin of the fBM plane and lines from T at the top of the sphere are then drawn to intersect both the sphere (at point I) and the fBM plane at each pixel point $(Xp, Yp) = P$ in the fBM plane with the value of P in the fBM plane being mapped onto I , on the surface of the sphere.

same index iv is then entered into the reference array block at the corresponding (Xd, Yd) point of the projection intercept with the display plane. If the reference array block already contains an index number, then another point on the planet's surface must have had the same projection point. Of course, the required point is that which is closer to the observer. So, using the data in the look-up table, a comparison of the distance from the observer to the surface point in question, for both the current and previous points, is made and the look-up table and display image array are then updated with the information relating to shortest vector.

ANTI_ALIASING AND COLONIZATION BANDING

The output of the display image can be structured many ways. In PLANET the illumination levels are normalized to provide a 0-255 gray scale, but their organization is switchable (switch *Icolor*) to provide a banded or a continuous gray scale range. A banded scale allows one to create special color effects based on the altitude of a surface point. In PLANET I chose to have three bands: one for the background, one at sea level, and one for all land masses. The background

was set to have a range of gray levels of 0-0, sea level (for $r = rsph$) was set for a range of 1-127, and the land masses (where $r > rsph$) were banded between 128 and 255. For example, consider the colored planet shown in Photo 4. Here the color assignments were black 0:0, blue 1:127, dark green 128:174, yellow 175:249, and white 250:255. If a continuous gray scale was chosen instead, the effect would be identical to that displayed by the moon (in Photo 4) where the illumination levels were normalized over a range of 0-255.

The image in Photo 4 was also corrected for simple aliasing effects by using a 1-D interpolation utility ap-

plied in the Y direction of the display image *disp* (switch *Int*). Its effect is best seen when *dis* is greater than *re* (i.e., when the projected image is magnified). It corrects for the ragged appearance of the image due to point density problems. But this 1-D interpolator is not sufficient and can experience problems. So, I suggest you write a full 2-D interpolation routine for best results.

AND FRACTALING ALONG

Well, I've run out of space and words again. But I hope I've given you some insight into the world of fractal computer art. In spite of the fact that the techniques described above are fun to use, they give us many insights into the highly variable and random mechanisms at work around us. As I said before, the conventional painter/artist needs only to create a personal interpretation of the visual representation of some form he sees in nature, and then render it on canvas. While we often find it necessary to know and understand the fundamental mechanisms from which its ultimate form is

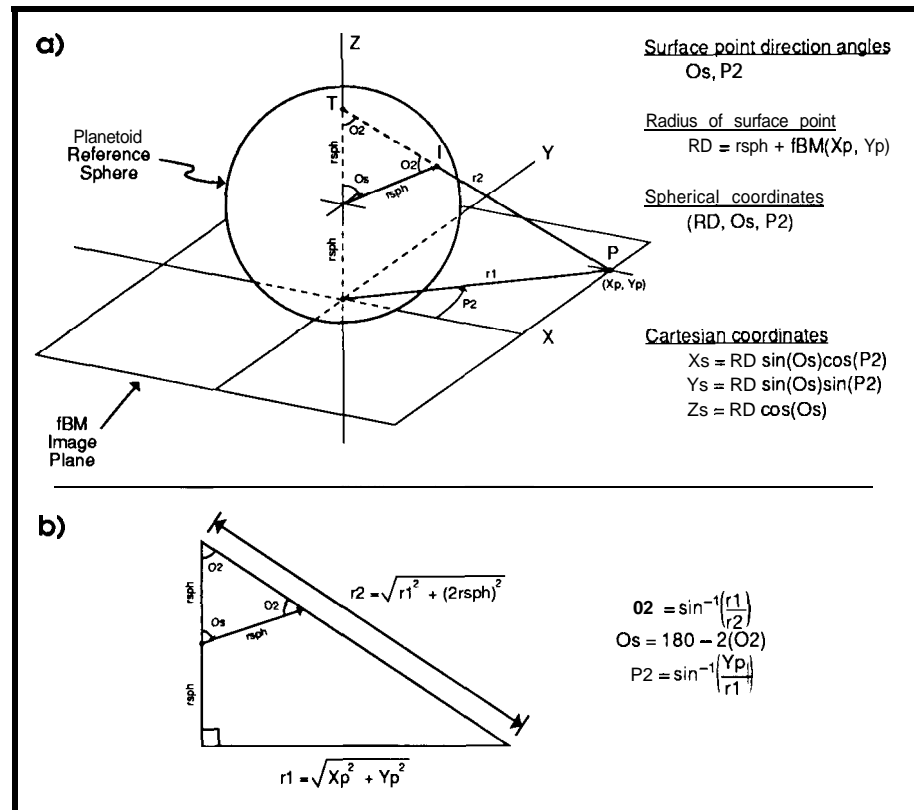


Figure 6—(a) Basic stereographic projection geometry used in the subroutine PROJ with (b) its associated orthogonal reference triangle.

derived, he creates images through simple experience and expression! We create images through experience, "understanding and modeling," and then expression! Have fun expressing yourself and creating your own fractal art. ❖

Chris Ciarcia has a Ph.D. in experimental nuclear physics and is currently working as a staff physicist at a national lab. He has extensive experience in computer modeling of experimental systems, image processing, and artificial intelligence. Chris is also a principal in Tardis Systems.

A commercial version of the software described in this article is available from:

Tardis Systems
945 San Ildefonso, Suite 15
Los Alamos, NM 87544
(505) 662-9401

REFERENCES

1. Mortenson, M.E., *Geometric Modeling*, John Wiley & Sons, 1985.
2. Mandelbrot, B.B. *The Fractal Geometry of Nature*, W.H. Freeman and Co., New York, 1983.
3. Lindenmayer, A., "Mathematical Models for Cellular Interaction in Development", Parts I and II, *Journal of Theoretical Biology*, 18:280-315, 1968.
4. Prusinkiewicz, P., "Graphical Applications of L-Systems", *Proceedings of Graphics Interface '86-Vision Interface '86*, pp 247-253, 1986.
5. Mandelbrot, B.B. and J.W. Van Ness, "Fractional Brownian Motion, Fractional Noises and Applications", *SIAM Review*, 10, 4 (October 1968), 422-437.
6. Peitgen, H.O. and D. Saupe, *The Science of Fractal Images*, Springer-Verlag New York Inc. 1988.
7. Voss, R.P., "Fractal Forgeries", in *Fundamental Algorithms for Computer Graphics*, R.A. Earnshaw, Ed., Springer-Verlag, (1985).
8. Lovejoy, S. and B.B. Mandelbrot, "Fractal Properties of Rain, and a Fractal Model" (EERM/CRMD, Meteorol. Nationale, Paris France) *Tellus Ser. A* (Sweden), vol 37A, no.3, pp 209-32 (May 1985).
9. Fraser, D.A.S., *Probability and Statistics: Theory and Applications*, Duxbury Press, Massachusetts, 1976.

IRS

253 Very Useful
254 Moderately Useful
255 Not Useful

Limited Editions

#G

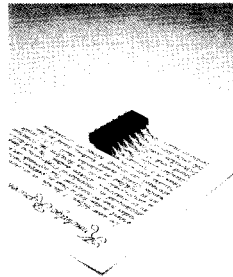
\$55



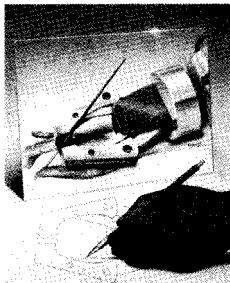
Programmable Hardware

#H

\$55



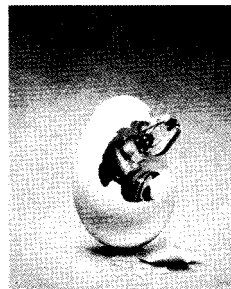
Word Processor



Intelligent Reflections

A

\$120



Technological Breakthrough

#C

\$70

Circuit Cellar Ink cover artist Robert Tinney proudly offers these distinctive 16" x 20" Limited Edition Prints. Each is an exquisite reproduction from the pages of *Byte Magazine*, and is part of an edition of only 1000 prints. The 100% cotton fiber stock is acid free, ensuring brilliance and durability for decades to come. The artist personally inspects, signs and numbers each print, which is accompanied by its own Certificate of Authenticity.

Order your Prints beautifully triple-matted and framed! The frames are of the silver metal variety, and mats are chosen to complement the colors of the print(s) you order. **plexiglass** only.

The price of each print is shown at left. Order two or more and deduct 15%! Frames are only \$39.50 each. For shipping, add \$4 per order for unframed prints (\$25 overseas); for framed prints, add \$5 for one print and \$3 for each extra Print (ground). No frames shipped overseas. Full refund if not satisfied. For VISA, MasterCard or AmEx orders, call 1-318-826-3003.

ORDER FORM

cci 1

Qty	#	Title	Amount
_____	_____	_____	\$ _____
_____	_____	_____	\$ _____
_____	_____	_____	\$ _____
_____	_____	_____	\$ _____
_____	_____	_____	\$ _____
If you order two or more, deduct 15%.			\$ _____
Frames (\$39.50 each)			\$ _____
Shipping charges: See above			\$ _____
Total			\$ _____

I have enclosed a check or money order to Robert Tinney Graphics. (Must be drawn on a U.S. Bank; no foreign collection, please.)

Bill my VISA MasterCard American Express account:

Card No.: _____ xpires: _____

Name: _____

Address: _____

City: _____ State: _____ Zip: _____

Country: _____

Send a brochure showing your other prints.

ROBERT TINNEY GRAPHICS
P.O. Box 778
Waxhington, LA 70589

FEATURE ARTICLE

J. Conrad Hubert

Running VGA on an IBM Professional Graphics Display

In a money-is-no-object computing scenario, I suppose everyone would have a 19" color monitor and a video card supporting 1280 x 1024 pixels in 256 colors along with a suitable CAD package on which to develop their projects. Unfortunately, most of us are unwilling or unable to spend \$5000 on a monitor and controller card, to say nothing of a CAD package capable of driving such a combination.

In my quest for cheap video, I was originally looking for a surplus 19" monitor which would do VGA. Why VGA? I have four CAD programs from three different vendors and, simply put, VGA is the highest common denominator. Beyond VGA resolution there are no "standards," and if the software does not explicitly support the display device you're out of luck.

My initial investigation into surplus 19" monitors showed them to be

plentiful, and extremely well made. When I considered that a VGA card and surplus monitor was an order of magnitude less expensive than the money-is-no-object scenario, they began to look very appealing indeed. The down side to surplus 19" monitors is that they are too heavy to lift alone, and lack "multisync" capability. This latter deficiency means that I must either be satisfied with analog EGA, which generates the same scan

rates as the 80 x 25 text screen, or "retune" the monitor each time I switched modes from EGA to VGA.

PROFESSIONAL GRAPHICS TO THE RESCUE

Then I discovered the IBM Professional Graphics Display. Although only measuring 13 inches diagonally, it is perfectly suited for VGA adaptation because it has a mode input which

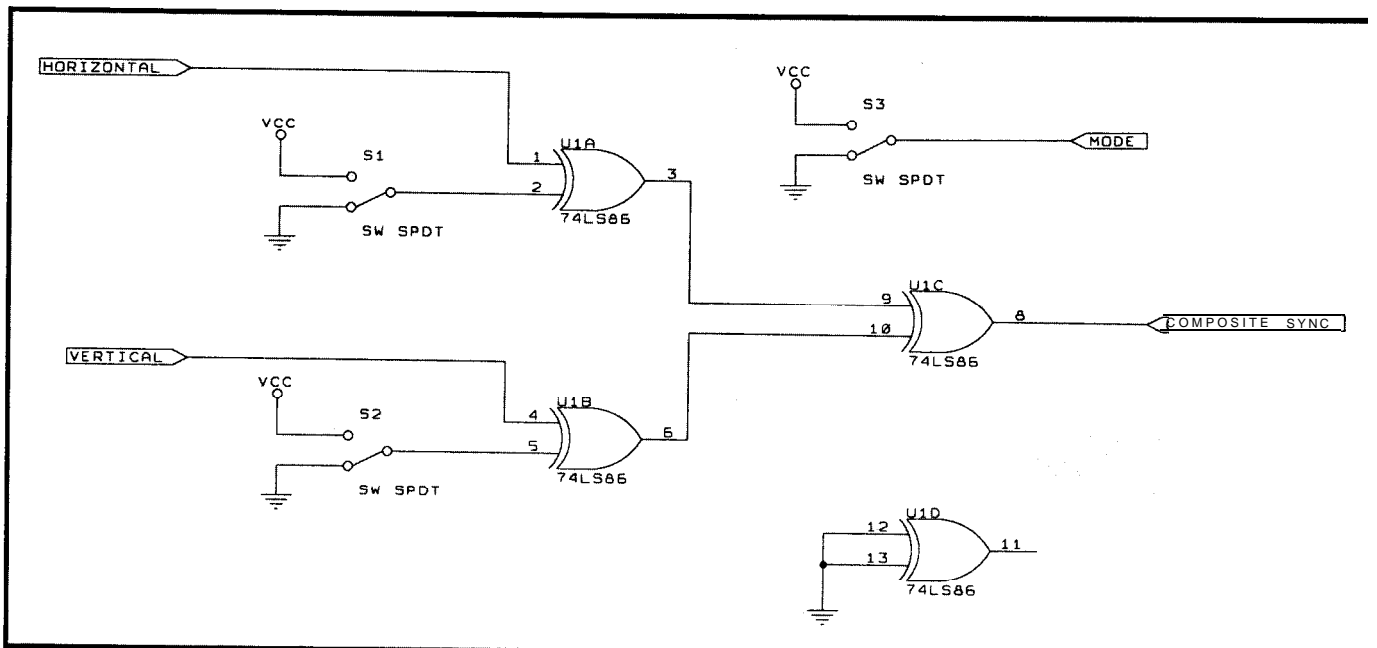


Figure 1—The VGA-to-PGA adapter circuit consists of just one chip and a few switches.

CAVEATS

The first, and most important caveat, is that once you step out of the 5-volt TTL world and into the **kV domain of CRTs**, certain precautions are necessary.

Rule 0. Remember, these things can **KILL** you!

Rule 1. Keep one hand in your pocket while making adjustments. (This reduces the **risk of current** flowing in one hand, through your chest, and out the other hand.)

Rule 2. Use only nonconductive diddle sticks.

The second caveat is that all electronic devices work on magic smoke, and once you let the magic smoke get out, they cease to function. One sure way to let the smoke out of a monitor's horizontal section is to operate it at a frequency higher than that for which it was designed.

allows two independent settings for vertical size. This permits an easy switch from EGA to VGA mode without retuning the monitor. For those of you unfamiliar with the so called PGA standard, it was IBM's answer to graphics-intensive computation in 1984. The 2-slot controller sandwich consumed 25 watts and boasted its own 8088 processor. Six years ago PGA was the rage, but times change and these boards are no longer considered a bargain. Consequently, their associated analog monitors can be obtained for next to nothing. (I paid \$75.00 for one and traded a broken Tandon computer for another.) They are beautifully built monitors and, with the modification described here, are ideal for VGA operation since the CRT shadow mask is the same as that of VGA- 640 x 480.

The only difficulty to overcome with the PGA monitor (and most surplus 19" monitors) is that they use a composite horizontal and vertical synchronization signal. In the VGA scheme, sync signals are separate. It's a simple matter, however, to "combine" these sync signals with a 74LS86

Everex ViewPoint mini-15 pinout (Other VGA cards may differ)

- 1 red
- 2 green
- 3 blue
- 4 reserved
- 5 ground
- 6 red return
- 7 green return
- 8 blue return
- 9 nc
- 10 sync return
- 11 nc
- 12 nc
- 13 horizontal
- 14 vertical
- 15 nc (On this pin I brought out +5V to power the 'LS86)

IBM PGA monitor DB-9 pinout

- 1 red video
- 2 green video
- 3 blue video
- 4 horizontal & vertical sync
- 5 mode select
- 6 ground for pin 1
- 7 ground for pin 2
- 8 ground for pin 3
- 9 ground for pins 4 & 5

Figure 2-The sync signals must be passed through the adapter, but the red, green, and blue signals run straight through.

<u>Mode</u>	<u>Horizontal Sync</u>	<u>Vertical Sync</u>
0	normal	normal
1	normal	inverted
2	inverted	normal
3	inverted	inverted

Figure J-The 'kind' of video being displayed is denoted by the polarity of the horizontal and vertical sync signals.

quad exclusive-OR gate as shown in Figure 1, while the red, green, and blue signals run straight through from the video board to the monitor (see Figure 2).

HOW DO YOU SYNC?

Have you ever wondered how a "mul tisync" monitor knows what type of video-CGA, EGA, VGA-it's getting? I did, and found that it's done via the sync signal's po-

larity. By inverting the horizontal and/or vertical sync signals shown in Figure 3, four different modes of operation are possible. I don't know what video board manufacturers call these

<u>Input A</u>	<u>Input B</u>	<u>Output</u>
L	L	L
L	H	H
H	L	H
H	H	L

Figure 4-The truth table for the exclusive OR operation shows how the A input determines whether or not the B input is inverted on the output.

<u>Video Format</u>	<u>Horiz. Sync</u>	<u>Vert. Sync</u>
Monochrome Text	normal	normal
16-color 640 x 350 (analog) EGA	normal	normal
16-color 640 x 480 VGA	inverted	normal

Figure 5—Three of the more popular display formats all use normal vertical sync, but one uses inverted horizontal sync.

modes, so I picked them arbitrarily to demonstrate the concept.

The aforementioned exclusive-OR gate also makes an excellent controllable complementer. Since there are three gates left in the '86 package, I used two of them for that purpose, and one is spare. These controllable complementers (possibly) invert the horizontal and vertical sync signals. When the A input for a given '86 gate is high, its output is the complement of the B input. Conversely, if the A input is low, the output is identical to the B input (see Figure 4).

INVERTING THE SIGNALS

Now it's just a matter of determining whether the signals need inverting, and diddling the pots inside

the PGA monitor to get the horizontal and vertical scan rates and sizes correct. I can give you what's in Figure 5, but after that you're on your own.

Of course, with this system, you do have to manually switch modes. This is possible to do automatically—I just haven't taken the time to do it since the optimal configuration for me is to use the PGA monitor for everything but CAD work and have a 19" monitor as a dedicated VGA display.

Since all of the above information is applicable to 19" analog monitors—you may want to experiment with them as well. I built a video "break-out" box to do just that. It consists of the 74LS86 circuitry, some toggle switches, a 3-terminal regulator, a 9-volt battery, and appropriate connectors. Concerning connectors—I should

point out that many 19" monitors use BNC jacks. This makes it easy to attach a 75-ohm coaxial cable, however in practice I've found a twisted pair works fine for 16-color video. I presume that with 256 colors the AV betweencolorsissmallenough that more care must be taken with the signals.

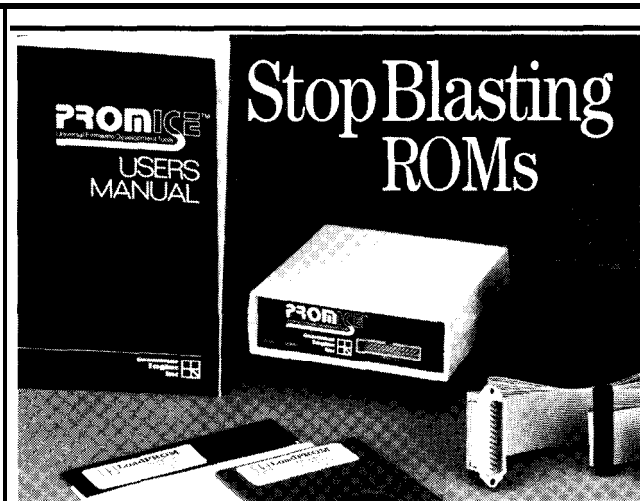
One final note of interest: I've found that Hitachi monitors from old Apollo workstations will run 1024 x 768, interlaced, from the Everex Viewpoint card. Although none of my CAD packages support this particular configuration, Microsoft Windows sure looked good! ❖

The author gratefully acknowledges Mr. Mark Kraemer for a discussion about the PGA standard.

J. Conrad Hubert owns Deus Ex Machina Engineering, a St. Paul, Minnesota consulting firm and is a partner in Silicon Alley Inc., a Seattle-based manufacturer of DSP products. In his spare time he likes to sleep.

IRS

- 256 Very Useful
- 257 Moderately Useful
- 258 Not Useful



Stop Blasting ROMs

PROMICE emulates 8 bit ROMs from 2716-27080, or 16 bit ROMs 27C1024 or 27C2048. (Inquire about emulating other ROMs. Non-JEDEC ROMs require custom cable.) ■ Sophisticated LoadICE™ Host Software downloads, uploads and edits ROM contents, supports MS-DOS, UNIX, MAC & VMS. Software sources are included. ■ Bi-directional Serial link, autobaud to 57.6KB—loads 1 Mbit in 25 secs. ■ Bi-directional Parallel port (option)—loads 1 Mbit in 4 sec. ■ Emulate up to 2 ROMs per unit, daisy-chain up to 256 ROMs from one port! ■ New! Analysis Interface™ (option) implements a ROM-based UART for sophisticated debugging.



1161 Cherry Street
San Carlos
California 94070
415/595-2252

CIARCIA'S CIRCUIT CELLAR, Vol. VII by Steve Ciarcia

Twelve new easy-to-build, cost effective, fun projects including:

- a gray-scale video digitizer...an infrared remote controller...the BCC180 Multi-tasking Controller...the Circuit Cellular IC Tester...the Smartspooler...an intelligent Serial EPROM Programmer...and much more.

214 pages.

CIARCIA'S CIRCUIT CELLAR

Volumes I-VII \$21.95 each

Call
(203) 875-2751 to order



Embedded Applications

43 Multichannel
Digital
Voltmeter
Interface
by Steve Garcia

58 Control Theory
for
Embedded
Controllers
by Thomas Mosteller

67 Using C
for
Embedded
Control
*by Ashok Patel &
Walter Banks*

Everything You Ever Wanted In UNIX. And Less. \$99.95*

OK. We know it's hard to believe. So just consider this. Coherent™ is a virtual clone of UNIX. But it was developed independently by Mark Williams Company. Which means we don't pay hundreds of dollars per copy in licensing fees.

What's more, Coherent embodies the original tenet of UNIX: small is beautiful. This simple fact leads to a whole host of both cost and performance advantages for Coherent. So read on, because there's a lot more to Coherent than its price.

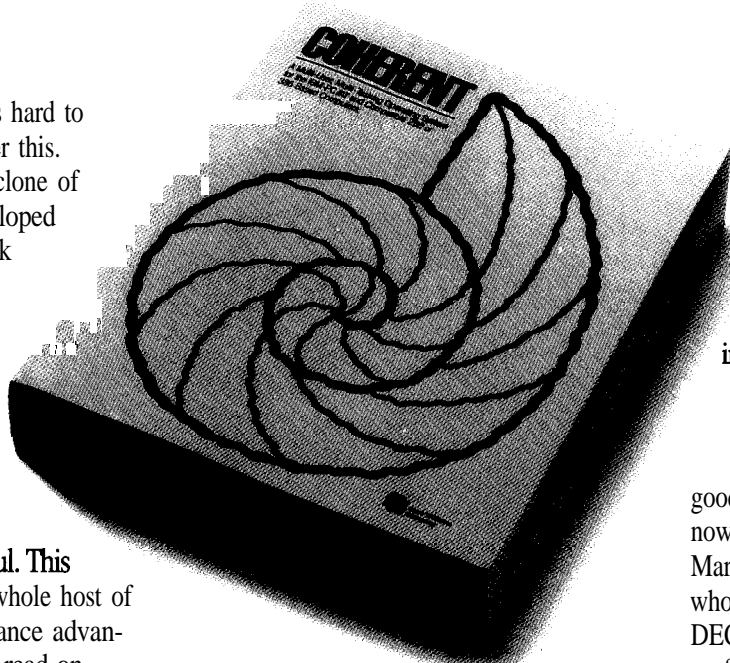
SMALLER, FASTER... BETTER.

Everybody appreciates a good deal. But what is it that makes small so great?

For one thing, Coherent gives you UNIX capabilities on a machine you can actually afford. Requiring only 10 megabytes of disk space,

LESS IS MORE!	Coherent For the IBM-PC/AT and compatible 286 or 386 based machines.	Santa Cruz Operation's XENIX 286, Version 2.3.2
No. of Manuals	1	8
No. of Disks	4	21
Kernel Size	64K	198K
Install Tune	20-30 min.	3-4 hours
Suggested Disk Space	10 meg	30 meg
Min. Memory Required	640 K	1-2 meg
Performance*	38.7 sec	100.3 sec
Price	\$99.95	\$1495.00

*3byte Exec benchmark, 1000 iterations on 20 MHz 386. Hardware requirements: 1.2 meg 5 1/4" or 1.4 meg 3 1/2" floppy, and hard disk. SCSI device driver available soon. Does not run on Microchannel machines.



Coherent can reside with DOS. So you can keep all your DOS applications and move up to Coherent. You can also have it running faster, learn it faster and get faster overall performance. All because Coherent is small. Sounds beautiful, doesn't it?

But small wouldn't be so great if it didn't do the job it was meant to do.

EVERYTHING UNIX WAS MEANT TO DO.

Like the original UNIX, Coherent is a powerful multi-user, multi-tasking development system. With a complete UNIX-compatible kernel which makes a vast world of UNIX software available including over a gigabyte of public domain software.

Coherent also comes with Lex and Yacc, a complete C compiler and a full set of nearly 200 UNIX commands including text processing, program development, administrative and maintenance commands.

And with UUCP, the UNIX to

UNIX Communication Program that connects you to a world-wide network of free software, news and millions of users. All for the cost of a phone call.

We could go on, but stop we must to get in a few more very important points.

EXPERIENCE, SUPPORT AND GUARANTEES.

Wondering how something as good as Coherent could come from nowhere? Well it didn't. It came from Mark Williams Company, people who've developed C compilers for DEC, Intel, Wang and thousands of professional programmers.

We make all this experience available to users through complete technical support via telephone. And from the original system developers, too!

Yes, we know \$99.95 may still be hard to believe. But we've made it fool-proof to find out for yourself. With a 60-day money-back no-hassles guarantee.

You have to be more than just a little curious about Coherent by now. So why not just do it? Pick up that phone and order today

You'll be on your way to having everything you ever wanted in UNIX. And for a lot less than you ever expected.

1-800-MARK WMS

(1-800-627-5967 or 1-708-291-6700)

60-DAY MONEY BACK GUARANTEE!



Mark Williams Company

60 Revere Drive
Northbrook, IL 60062

*Plus shipping and handling. Coherent is a trademark of Mark Williams Company. UNM is a trademark of AT&T. XENIX is a trademark of Microsoft.

Multichannel Digital Voltmeter Interface

FEATURE ARTICLE

Steve Ciarcia

MAX734 Chip Adds High-Performance ADC to Embedded Control

When I left you last time I had to apologize that the project I expected to present turned out to be something entirely different. I had fully expected that instrumenting my new solarium would be the perfect design to describe the latest in domestic computer applications. As it turned out, a thermostatically controlled vent fan adequately regulated the temperature; a fact that we verified with the data logger I presented instead. This was not a surprise. It was just good engineering.

Whenever an engineer turns a paper project into reality there are adjustments. I make a joke about the fact that I put together an application for embedded control and then left out the control but that is not quite correct. In truth, mechanical thermostats and solar-activated vent windows do indeed constitute a control system. The difference is that I didn't need the addition of an embedded computer to effect adequate control in this situation.

DÉJÀ VU

I ended the data logger project by suggesting that, while my solarium couldn't justify computer control, surely if I built a freestanding green-

house, it would definitely need it. Reality often hurts, but here are the facts.

We built an 11- x 20-foot redwood and glass greenhouse early this spring. The roof is double-paned glass with translucent plastic thermoshielding and four 1- x 5-foot solar-powered

control system (HCS), to give the plants a little music. So far my wife has pushed about 3000 plants through this greenhouse and it hasn't been empty since the last pane was installed.

Most of the flowers passing through the greenhouse start out as 100-300 "plugs" in 24" x 14" trays. The individual plants are then transplanted into 6" x 6" six-plant containers or into 4" to 6" pots. Presently, there are 250 chrysanthemums in 6" pots filling the place (I can't wait to see where she's going to put them when they are ready to plant in the yard).

Watching this massive growth and production of greenery has given me enough infor-

mation to formulate a few axioms. First, if you have someone in the house who likes plants but has been happy with a few window pots, be careful about instantly upgrading to a full greenhouse. Instead of picking up a couple 20-lb bags of potting soil at K Mart anymore, you suddenly find yourself ordering pickup trucks full of bags and coming home to find dump trucks unloading piles of brown and black stuff for new flower beds. Instead of a few plants picked up on the way home from the local nursery for pocket change, everything graduates



Photo 1 -The Circuit Cellar greenhouse is next in line for computerization.

vents. As the temperature rises, a chemical inside the vents' cylinder-shaped actuators expands allowing a spring to open the vent. As the greenhouse cools, the vents close.

Just these and a door window vent were adequate until June. Then, to supplement the cooling, I mounted a thermostatically controlled attic fan and louvered vent in the back wall. When the temperature exceeds about 80°F (or wherever I set it to), the fan automatically comes on. I even installed a radio, controlled through the power line from the existing home

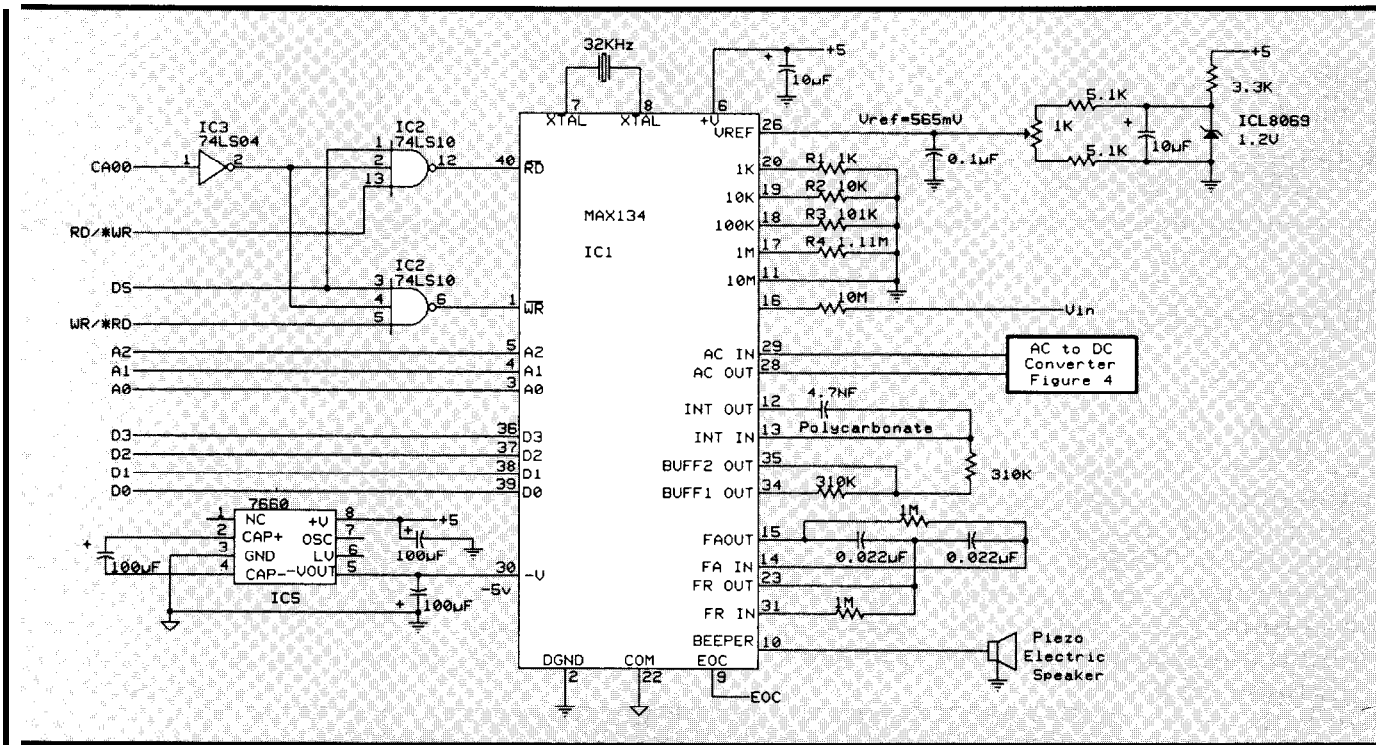


Figure 1 - \bar{m} e BCC bus digital voltmeter interface is based on the MAX134 chip and can measure 400 mV-4000 V AC or DC.

to wholesale accounts and Federal Express deliveries of hundreds of plants.

In reality, I shouldn't complain. The place looks beautiful and my wife is having fun. She could be into electronics or computers after all.

The fact that we have a real working greenhouse did allow me to see what was needed. When I first thought of the greenhouse as an embedded-controller paradise, I thought of a year-round operation incorporating automatic watering, automatic fertilizing, humidity control, venting, cooling, and lighting. What I discovered since then is that four solar vents and a thermostatically controlled fan can accomplish 90% of the environmental control from spring through fall (forget humidity control in an open greenhouse) for about 10% of the potential expense. While I do have a 50,000-BTU propane heater for winter, the projected cost of using it is scary. Experiment or not, I don't think I can justify \$300-\$500 a month to warm wax begonias and dwarf morning glories when I can move them into the "inside greenhouse" for the winter (did I mention the 8' x 20' greenhouse on the side of the solarium?). It would take growing a cash crop like orchids before I could justify heating all year.

Automatic watering seemed to be about the only remaining candidate.

If you have large mature plants, potted varieties contained together, or distinctly individual watering requirements, separate feeding and watering tubes might be advantageous. However, if instead you have a thousand plants in containers resembling ice cube trays, only a wide-area overhead spray can water them adequately. Ringing the greenhouse with 1" plastic pipe and spray heads and using a lawn sprinkler timer could make watering automatic but at what price? We aren't talking about a 40' x 100' greenhouse. Spray watering here sounds too much like make-work control to me. (Besides, my wife enjoys going out there and watering each morning.)

The application of automatic controls in a greenhouse is solely dependent upon the size of the greenhouse and the value/effort quotient of the product being grown. Commercial greenhouses need spray watering for plant trays and drip watering for hanging baskets simply to save labor (it's a five-minute job in our greenhouse). The addition of pH measurements, automatic fertilizing, CO₂ injection, and other special environ-

mental controls are generally justified only as an experiment, to maintain consistent growth of valuable plants, or to simulate an environment inside the greenhouse which is more propitious than the environment outside. Petunias don't fall into this category.

A NEW CHALLENGE

Considering my beautified yard, I wasn't completely unhappy about the greenhouse, but it did appear a little like solarium *deja vu*. While the goal was accomplished, the experiment was yet to be attempted.

Rather than trash the whole idea, I decided to look at the task from a different point of view. It didn't take long to conclude that my experiment wasn't bold enough. My failure was not thinking futuristically. I didn't need a computer-controlled greenhouse for petunias, but certainly some esoteric agricultural species must.

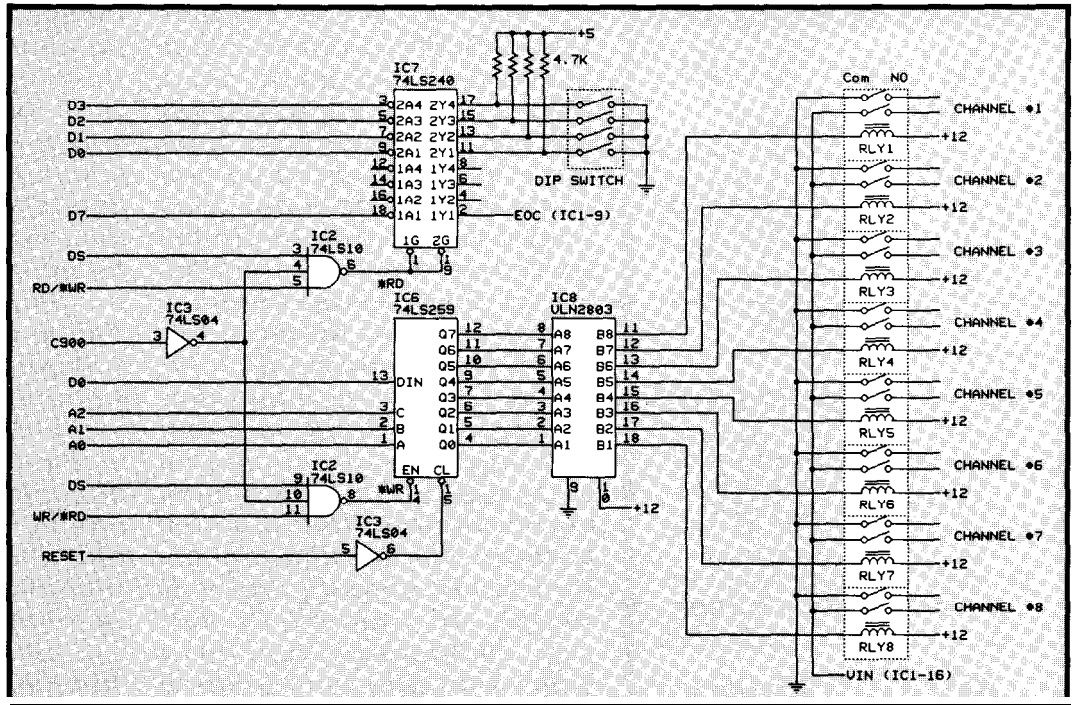
A few weeks ago I read an article in the local newspaper about a Connecticut farmer who was growing giant tomatoes hydroponically. I was so intrigued by the idea that I had hardly finished the article before jumping into the car to see this stuff close up.

Three hours later we were on the other side of the state pulling up to the two 36' x 100' greenhouses mentioned in the article. Fortunately we were not greeted with a shotgun.


After I introduced myself and explained that I wasn't a spy, events progressed rapidly. My host could best be described as a high-tech gentleman farmer. His only agricultural endeavor

was hydroponically grown tomatoes and he fully intended to corner the market. He thoroughly understood the potential of embedded controls and monitoring but said that, outside of

Figure 2—In order to maintain the high degree of electrical isolation provided by the MAX134, simple mechanical relays are used to multiplex the input to the chip. When added to the circuit in Figure 1, this circuit increases the DVM capacity from one to eight channels and also supplies a user DIP switch interface.



PC BIOS Development · Industrial Control · Embedded Systems · ROM Development



ROM
EEROM
SRAM
SRAM AS ROM

The Kolod Research **R³OM Card**


SUPPORTS MOST 28-PIN JEDEC MEMORY DEVICES. YOU CAN USE STATIC RAMS AS ROMS TO DYNAMICALLY DEVELOP AND TEST PC ROM BASED CODE *WITHOUT* BURNING IN ROMS AND FULLY DEBUG THE ROMS YOU DO BURN

The R³OM card provides all the facilities you need to develop and deliver PC based BIOS, embedded systems, Industrial control software, and ROM based applications

the R³OM card features include:

- 4 independent 28 pin JEDEC defined sockets, each socket's address is switch-selectable to anywhere within the 0 to 1 Mbyte address range
- ability to address and configure sockets independently or consecutively for a total of upto 128K SRAM, 256K ROM, or any combination of ROM, EEROM or SRAM in between
- use only the sockets and memory you need
- jumper select battery backup for any of all of the sockets
- variable wait-state generation for each of the sockets - means you can use slower memories in high-speed systems
- software or hardware controlled write protect for each socket (for SRAMs and EEROMs)
- unique software driven 3 channel DMA fly-by test circuitry
- flexible port I/O addressing
- complete technical documentation including programming and hardware specifications with examples (source code included, of course)
- full support for PC/XT/AT/386 machines
- high quality multi-layer construction

Engineering Excellence

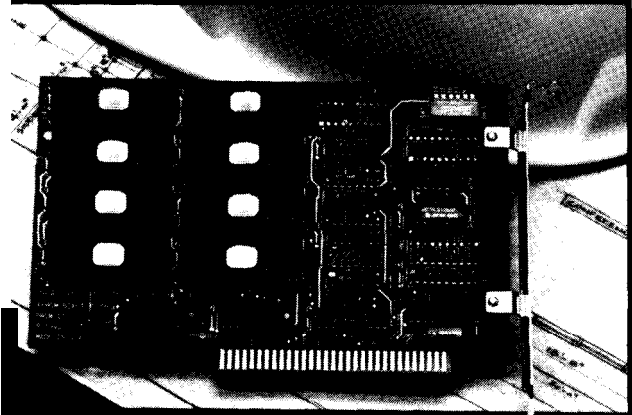


Kolod Research Inc.
1898 Techny Court, Northbrook, Illinois 60062-3474 U.S.A.

\$285.00 - shipping/handlin
Made in U.S.A

(312) 291-1586 for other options, sales & technical information

Reader Service #150



PROM-III

- PUT DOS AND APPLICATION IN EPROM
- ALLOWS DISKLESS OPERATION
- UP TO 1 MBYTE ROM-DRIVE WITH 16K FOOTPRINT
- PROMKIT SOFTWARE BY ANNA-BOOKS
- FLASH EPROM SUPPORTED
- BATTERY RAM MODULES SUPPORTED
- DELIVERY FROM STOCK

SEALEVEL
COMMUNICATIONS & I/O

SEALEVEL SYSTEMS, INC.
PO BOX 1808
EASLEY, SC 29641
[803] 855-1551

Reader Service #184

8031/51 Tools

8031 In-Circuit Emulation

Our emulator provides most features of an 8031 In-Circuit-Emulator at a significantly lower price. It assists in integration, debug, and test phases of development. Commands include: disassembly, trace, breakpoint, alter register/memory, and load Intel Hex file. **\$199**

8051 Simulation

The 8051 SIM software package speeds the development of 8051 family programs by allowing execution and debug without a target system. The 8051SIMulator is a screen oriented, menu command driven program doubling as a great learning tool. **\$99.**

8031/51 Single Board Computer

A fast and inexpensive way to implement an embedded controller. 8031/32 processor, 8+ parallel I/O, up to 2 RS232 serial ports, +5 volt operation. The development board option allows simple debugging of 8031/51 family programs. **\$99ea**

Prototyping System

The IPC-52 development system allows you, the designer, to concentrate on Application Specific Circuitry only, because the 8052, RAM, EPROM, and I/O sections are built-in and fully functional. The prototyping bread-board is integrated into the system - save days of development time. **\$220**

Call us for your custom product needs.

Other products available:

MyGAL - GAL Programmer **\$199**

FORTH Card - FORTH development card for STD Bus **\$279 (OEM-\$1 99)**



HiTech Equipment Corp
9400 Activity Road
San Diego, CA 92126
(FAX: (619) 530-1458)

(619) 566-1 892

experimental greenhouses, commercial hydroponics was only semiautomatic at most.

Stepping into his greenhouse was an experience. Calling these "tomatoes," like the ones you see in a grocery store, was an understatement. These TOMATOES were the size of small melons. The tomato plants, which lived 2-5 years, were 12 feet long with 1.5" diameter stalks. Ropes strung from overhead frames supported the plants which were heavy with tomatoes hanging in bunches like giant grapes. I wouldn't be surprised to count 100 tomatoes on a single plant considering he was harvesting 1000 lbs per week per greenhouse.

The wildest thing about hydroponics is the growth medium. Two of these giant plants were growing out of a two-inch-thick one- by four-foot bag of rock wool. Where each stalk protruded from the rock wool bag, a watering tube dripped a mixture of nutrients and water onto the plant roots. All the tubes plugged into a pipe which came from a central "mixing" station. There, an assortment of pumps and valves channeled pre-mixed fertilizer and water to the plants.

The regulated introduction of the nutrients and water is the secret to hydroponics. Of course, as I have determined subsequently from fertilizer company literature, the ingredients are not really secret, but just like soil gardeners, the farmer who pays absolute attention to detail will have better produce. Fortunately, those of us who understand embedded control know that there is no better method of "absolute attention." Perhaps we can show these farmers a thing or two.

When I returned home and started collecting information on hydroponic gardening it became readily apparent why more people aren't doing it. The mixtures aren't secret all right, but you have to have a lot of experience or a chemical engineering degree. In hydroponics, one doesn't just mix a few tablespoons of Miracle Gro and pour it on the plant. Instead, they mix separate barrels of calcium nitrate and Hydrosol with a few trace elements and then add sulfuric acid or potassium hydroxide to adjust the pH to a

constant 5.7 acidity. This is while you maintain 64 ppm of this and 200 ppm of that and so on. Oh yeah, be careful your iron doesn't precipitate out if you add things in the wrong order.

To add insult to injury, this brew changes both for the kind of plant and for its stage of growth. Cucumbers and tomatoes, for example, don't seem to eat the same thing if you want maximum production of each. A brew heavy in nitrates designed to promote the early stage growth of one plant can stunt the growth of another. Add to the mixing task a job of constantly monitoring and adjusting the specific ion concentrations with the additional environmental effects of temperature variation and evaporation and it's easy to go back to growing petunias.

I am not promising to grow hydroponic tomatoes, but I am intrigued. Looking at this high-tech, but still relatively manual, operation made me wonder if we could computer control the whole mess cost-effectively. Using readily available flow control valves, pH and temperature sensors, metering pumps, mixing tanks, and a little software we should be able to configure a simple recipe system.

MAKING A PLAN

Considering the complexity of such a project, a rational game plan is necessary. Hydroponics appears to me as basically a combination of mixing, monitoring, and timing. As we already know, computers are very adept at timing, and with the addition of common output drivers they can easily operate valves and pumps. The BCC52 and BCC180 controllers, which are available with a variety of control outputs, are more than adequate in this application.

The only real unknown in this venture is the level of complexity and performance required of the input sensors which will monitor the various analog parameters such as concentrations, temperature, pH, and humidity. While a two-degree temperature resolution may have been adequate in the solarium, proper calculation of concentrations to a few parts per million (ppm) will require

considerably more precise measurement. The pH measurement is also a big unknown at this time. The sensors seem to have resolutions of 0.1%. Do we have to maintain such accuracy or is it only relative. Obviously we can't monitor it grossly and then conclude that precise monitoring is necessary.

ONE MULTIRANGE ANALOG INPUT INTERFACE

Since I have absolutely no idea what I will encounter for sensor outputs until I get into hydroponics more deeply, my only choice was to design an analog-to-digital control interface which had a very wide range as well as high accuracy.

Measuring millivolts to hundreds of volts on the same input channel is no easy task. Most high speed A/D converters used in embedded controllers are 8-12 bits and have a useful range of 0-5 V or -5 V to +5 V. Sacrificing range for speed, such units are wholly inadequate at accurately measuring 100-mV or 100-V signals.

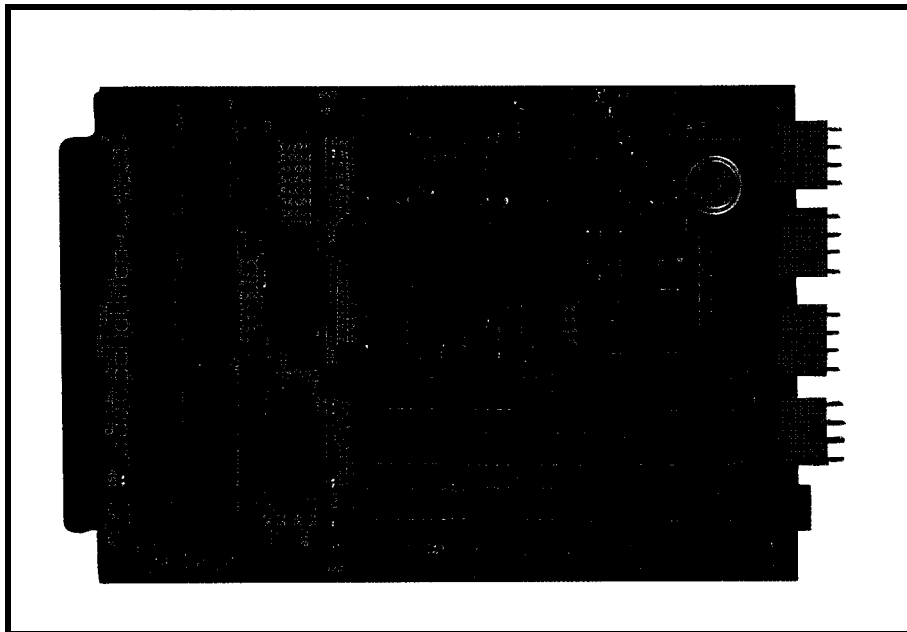


Photo 2-A BCC-bus prototyping board was used to make interfacing the MAX134 to the rest of the system easier.

The best method for measuring low-level DC inputs is to use slower integrating A/D converters instead. Integrating ADCs are typically used in digital voltmeters.

If you've ever used a digital voltmeter I shouldn't have to convince you of its performance. Even the lowly 3.5-digit units can accurately read a 100-mV input on their 0-199.9 mV

BCC52 BASIC-52 COMPUTER/CONTROLLER

The BCC52 Computer/Controller is Micromint's hottest selling stand-alone single-board microcomputer. Its cost-effective architecture needs only a power supply and terminal to become a complete development or end-use system, programmable in BASIC or machine language. The BCC52 uses Micromint's 80C52-BASIC CMOS microprocessor which contains a ROM-resident 8K-byte floating-point BASIC-52 interpreter.

The BCC52 contains sockets for up to 48K bytes of RAM/EPROM, an "intelligent" 2764/128 EPROM programmer, three parallel pods, a serial terminal port with auto baud rate selection, a serial printer port, and is bus-compatible with the full line of BCC-bus expansion boards. BASIC-52's full floating-point BASIC is fast and efficient enough for the most complicated tasks, while its cost-effective design allows it to be considered for many new areas of implementation. It can be used both for development and end-use applications.

Processor

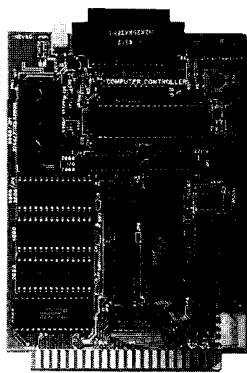
- 80C52-BASIC, 8-M CMOS microcomputer
- jumper-selectable conversion to 80C31/80C32 functionality
- 8K bytes ROM (MI BASIC interpreter)
- 256 bytes RAM
- three 16-bit counter/timers
- 32 I/O lines
- 11 MHz system clock
- 6 interrupts

Memory

- expandable to 62K bytes
- five on-board sockets
- up to four 6264 (8Kx8) static RAM
- either 2764 or 27126 EPROM

Input/Output

- console I/O RS-232 serial port
- line printer RS-232 serial port
- three 8-bit programmable TTL-compatible parallel I/O ports using a 8255 PPI
- alternate console RS-422/RS-485

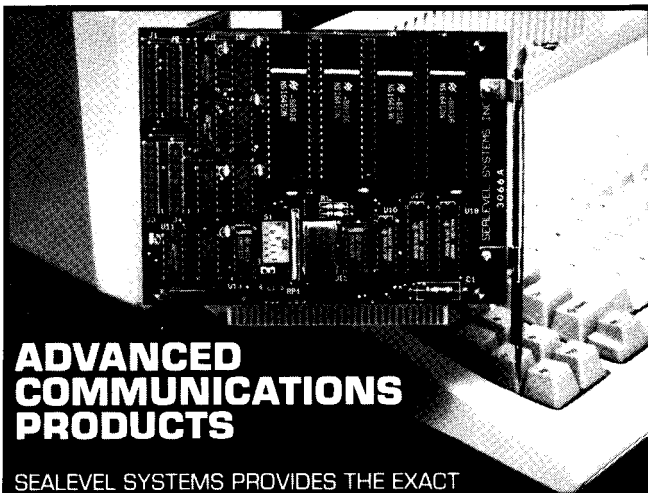


To Order Call
1-800-635-3355
Tel: (203) 871-6170
Fax: (203) 872-2204
TELEX: 643 3 3 1

		Single Qty.	100 Qty.
BCC52	BASIC-52 Controller Board with 8K RAM	\$189.00	\$149.00
BCC52C	Lower-power all-CMOS version of the BCC52	\$199.00	\$159.00
BCC52I	Full industrial temperature range	\$294.00	\$220.00
BCC52CX	CMOS, Expanded BCC52 w/32K RAM	\$259.00	\$159.00

MICROMINT, INC. 4 Park Street, Vernon, CT 06066

Reader Service #162



ADVANCED COMMUNICATIONS PRODUCTS

SEALEVEL SYSTEMS PROVIDES THE EXACT COMMUNICATION CARDS YOU NEED.

PRODUCTS:

- 1, 2 OR 4 PORT RS-232 AND RS-422/485 BOARDS
- CURRENT LOOP SERIAL INTERFACES
- HIGH SPEED SYNC (HDLC, SDLC) AND ASYNC WITH DMA
- RS-530 AND V.35 INTERFACE BOARDS
- DIGITAL AND RELAY I/O BOARDS
- DISKLESS EPROM BOARD WITH PROMKIT SOFTWARE BY ANNABOOKS
- NEW LAP-TOP ADD ONS!
- DELIVERY FROM STOCK
- MADE IN USA
- SATISFACTION GUARANTEED
- EXCELLENT TECHNICAL SUPPORT



SEALEVEL SYSTEMS INC.
PO BOX 1808
EASLEY, SC 29641

[803] 888-1881

Reader Service #185

range; that's a resolution of 0.1 millivolts! If the input rises to 1 volt then we can switch to the 0-1.999 V or 0-19.99 V scales. Should the input shoot to 100 V there's even a 0-199.9 V (and for the brave a 0-1999 V) scale. Smart voltmeters even do the range switching automatically.

Since temperature, pH, battery levels, and so on won't change all that rapidly, it would seem that an interface that functions like a digital voltmeter would most assuredly provide the wide analog acquisition requirements for our control system. An analog input board with an integrating analog-to-digital converter is the perfect interface solution.

THE MAX134 DIGITAL MULTIMETER CHIP

Figures 1 and 2 contain the schematic of the DVM interface board I finally arrived at to solve my problem (shown in Photo 2). It can be built in various configurations depending upon the application (more later) but, as shown, the DVM board has eight

differential input channels which can be independently programmed to read any AC or \pm DC input within the following ranges: 0-399.9 mV, 0-3.999 V, 0-39.99 V, and 0-399.9 V (and more). The DVM board circuitry is assembled on a BCC55 BCC-bus pro prototyping card for convenience.

The heart of my DVM interface board is the Maxim MAX134 digital multimeter chip. The MAX134 is a bus-compatible chip which is designed specifically to interface with a microprocessor to perform autoranging, scaling, autozeroing, and function switching. The MAX134 has three address lines, two control lines, and four data lines. Numerical readings are read directly as BCD digits.

The MAX134 ADC's internal resolution is really $\pm 40,000$ counts even though it is referred to as a $3\frac{3}{4}$ -digit converter. In a direct comparison with other ADC chips, the MAX134 really has about 16 bits of resolution. All five digits can be read and displayed, however the least-significant digit is generally used as a guard digit in autozero programs and is not neces-

sarily displayed. Ultimately, it will be component layout, grounding, and noise shielding on your board which will determine whether you choose to display a 101+ millivolt input as 101.4 (13 bits) or 101.43 millivolts (16 bits).

Block diagrammed in Figure 3, the MAX134 is designed to be a full-function multimeter chip. The MAX134 provides all the logic and counters for control of the conversion and the external processor does not have to perform any critical timing. The MAX134 includes 26 internal switches to selectively gate the connection of components and the flow of current through the circuit. Set by values poked into registers by the processor, these switches gate five decades of attenuation and mode-select the circuitry for AC or DC, voltage, current, ohms, autozero, and continuity measurement. Oh yes, it can beep too.

RANGE SELECTION

Shown in Table 1, these control bits set the open and closed position of the internal gates. Range selection is an easy example of how they work. Basically, the MAX134 is a 400-mV full-scale A/D converter. Voltages are applied through a 10-megohm resistor and then divided through a shunt resistor network (ideal values are 1.1111 M, 101.101 k, 10.010 k, and 1.0001 k ohms) until the input voltage is within the 400-mV scale.

The 10-0 bit selects the 10M input (pin 16) without activating any shunt resistors. This sets a 0-400 mV input range (if you are only going to use the 400-mV range and no others, the MAX134 has a single 400-mV scale input pin which would be used instead). The 10-1 bit activates the 10:1 attenuator and sets the 0-4 V range by selecting the 10-M Ω input resistor and the 1.1111-M Ω shunt. Similarly, the 10-2, 10-3, and 10-4 bits select input attenuation factors of 100, 1000, and 10,000 respectively. This last attenuator suggests this interface has a 0-4000 V range. For reasons of safety and ratings of other components in the system, however, I have specified this interface as 0-400 V even though the attenuation resistor is in the schematic.

HURDLE MICROCONTROLLER BOUNDARIES *with* **Byte-BOS™** Integrated Systems REAL-TIME MULTITASKING OPERATING SYSTEM

Byte-BOS Real-Time Multitasking Operating System (BOS) is a powerful multitasking operating system designed specifically for embedded microcontroller applications.

BOS is written in "C" with an assembly language kernel tuned to a specific microcontroller. Application code written in "C" using BOS on one microcontroller can be used on any other microcontroller supported by BOS.

BOS reduces integration time by supporting "on board" peripherals and popular "C" compilers. BOS includes timer support, an asynchronous communications package, system "make" utility and working application code.

BOS supports a wide variety of microcontrollers including the 8051, 8096, 80188/188, 6801/3, 68HC11, 68332, 68302, 68340, 630113, 64180, H8500, and 37700 families.

BOS is also available for the PC environment and will work with DOS. The BOS PC system can be used in a desktop of embedded application and can also be used to develop software for other microcontrollers supported by BOS.

BOS is available as "no royalty" source code. The complete system sells for \$1990 and includes a user manual, "make utility", and application code. BOS supports one timer and serial port, and is configured to the "C" compiler of choice.

Byte-BOS™
Integrated Systems

415-543-3626

The MAX134, like other A/D converters, is a DC-only device. To read AC voltages they must first be converted from AC to DC. In the case of the MAX134, this is again accomplished by setting the register bits controlling the internal switches. When an AC value is applied, the DC attenuators are set to acquire the proper range, but instead of sending the voltage directly to the filter section and ADC, the EXT AC bit channels this voltage out to an external AC-to-DC converter.

Figure 4 is the circuit of the AC-to-DC converter which ultimately used. The original circuit built and pictured on the prototype board was just that: a prototype. Figure 4, an improved design, is a typical half-wave AC-to-DC converter. The output is proportional to the average AC value rather than the RMS value (you could use a RMS converter, but this is the method typically used in digital multimeters). It is easily calibrated using the gain and offset adjustments. Simply apply an AC voltage, such as 2.00 VAC as

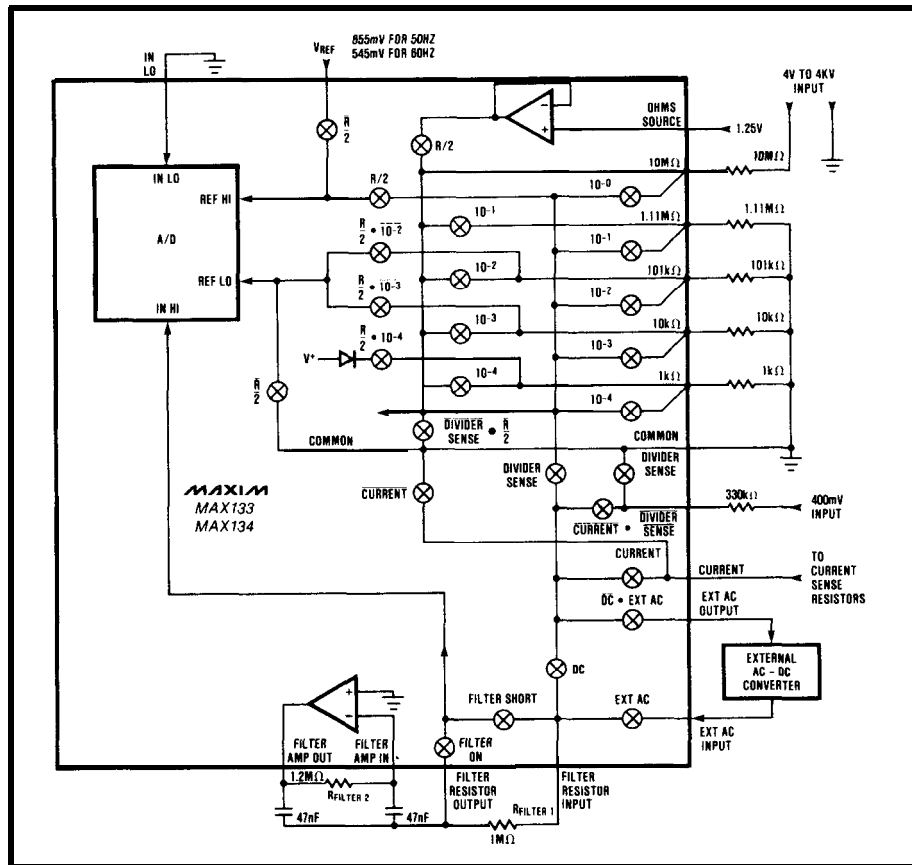


Figure 3—The MAX 134 is designed to be a full-function multimeter chip.

NEW!!
68000, COP800, PIC16xx
versions!

µASM™ Cross Assemblers
for the Macintosh™

- *TEXT EDITOR, CROSS ASSEMBLER, AND COMMUNICATIONS FACILITY IN A COMPLETE INTEGRATED DEVELOPMENT ENVIRONMENT
 - MACROS
 - CONDITIONAL ASSY
 - LOCAL/AUTO LABELS
 - SYMBOL TABLE CROSS REF
 - S OR HEX FILE OUTPUT DOWNLOADS TO MOST EPROM PROGRAMMERS
- us **\$149.95**
EACH
PLUS S/H*

AVAILABLE FOR MOST 8-BIT MICROPROCESSORS AND 68000/010. CALL OR WRITE FOR TECHNICAL BULLETIN. **30 DAY MONEY BACK GUARANTEE.** MC/V/AE.

Micro Dialects, Inc.
DEPT. C, PO BOX 30014
CINCINNATI, OH 45230
(513) 271-9100

• PER SHIPMENT:
\$4 CONTIGUOUS USA
\$8.50 CANADA AK, HI
\$15 INTERNATIONAL

FREE CATALOG
Tools & Test Instruments



Packed with thousands of important, up-to-date products for testing and repairing electronic equipment. Products are shown in full-color with detailed descriptions and pricing.

- Digital Multimeters
 - Oscilloscopes
 - Soldering Supplies
 - Electronic Tools
 - Tool Kits
 - Static Control Products
 - Telecom/Datacom Instruments
 - EPROM Programmers/ Erasers
- Plus much more...

In a hurry? CALL **(508) 682-2000**
Contact East, Inc., Dept. 2433,335 Willow St., No. Andover, MA 01845

INTROL

CROSS DEVELOPMENT SYSTEMS

- INTROL-C Cross-Compilers
 - INTROL-Modula-2 Cross-Compilers
 - INTROL-Macro Cross-Assemblers
- provide cost and time efficiency in development and debugging of embedded microprocessor systems

All compiler systems include:
 Compiler • Cross-assembler • Support utilities • Runtime library, including multi-tasking executive • Linker • One year maintenance • User's manual, etc.

TARGET3 SUPPORTED:
 6301/03 • 6801/03 • 6804 • 6805 • 6809
 • 68HC11 • 68000/08/10/12 • 32000/
 32/81/82 • 68020/030/881/851

AVAILABLE FOR FOLLOWING HOSTS:
 VAX & MicroVAX; Apollo; SUN; Hewlett-Packard; Gould PowerNode; Macintosh; IBM-PC, XT, AT and compatibles

INTROL CROSS-DEVELOPMENT SYSTEMS are proven, accepted, and will save you time, money, effort with your development. All INTROL products are backed by full technical support. CALL or WRITE for facts NOW:



INTROL

CORPORATION

647 W. Virginia St., Milwaukee, WI 53204
 414/276-2937 FAX: 414/276-7026
 Quality Software Since 1979

Reader Service #125

Register Map of Input Data From the Microprocessor to the MAX134

Address or Register Number	D3	D2	D1	D0
0	Hold	High Frequency	Beeper ON	Sleep
1	1 0-0	Filter Short	+5	50Hz
2	10-4	10-3	1 0-2	10-I
3	DC	Ext AC	Divider	Ohms R/2
4	Current	X2	Sense Read Zero	Sense Filter On

DESCRIPTION OF CONTROL BITS

Hold: A 1 in Hold will stop conversions at the end of the next conversion. If the MAX 133/134 is in the Hold mode, a conversion will start on the next clock cycle after Hold is set to 0. The oscillator continues to run and all circuitry is active during the Hold mode.

High Frequency: A 1 in the High Frequency bit will select 4096 Hz as the beeper frequency. A 0 will select 2048 Hz.

Beeper On: A 1 turns on the beeper driver.

Sleep: A 1 in Sleep puts the MAX134 into the standby or sleep mode. The Common voltage buffer is turned off and the internal analog circuits are turned off, but the DGND circuitry is still active. The oscillator continues to run. Current consumption is reduced to 25 μ A. Several conversions must be performed after exiting the Sleep mode before full conversion accuracy is obtained.

10-0 through 10-4: These bits control the attenuator network switches. The 10-0 bit selects the 10M input without activating any shunt resistors. This is an alternate 400-mV input. The 10-1 bit activates the 10:1 attenuation by selecting the 10M input and connecting the 1.111 M shunt. Similarly, 10-2, 10-3, and 10-4 bits select input attenuation factors of 100, 1000, and 10,000 respectively. In the ohms mode these bits set the resistance range.

50Hz: When set to 1 the integration period for voltage measurement is one cycle of the 50-Hz power line (655 clock cycles). When 0, the integration period is one 60-Hz power line cycle (545 dock cycles).

X2: Setting the bit to 1 activates the MAX134 "times 2" function. When X2 is active, Rint2 only is used as the integrator resistor during the integration phase. Rint1 and Rint2 in series are used as the integration resistor for all deintegration phases and for the integration phase when X2 is 0. If Rint1 = Rint2 then setting the X2 bit doubles the digital output for a given input voltage.

Divide by 5: When this bit is set to a 1 the integration period is reduced by a factor of 5. This reduces the digital output code by a factor of 5, and allows a higher input voltage to be used. The full scale input voltage is multiplied by 5 when this bit is set, but caution should be used to make sure that the 2- μ A maximum recommended integrator output current is not exceeded, or the MAX134 linearity will be degraded.

Ohms or R/2: Setting this bit to a 1 causes the next conversion to be a Read Zero conversion. A read zero conversion is performed with In Hi and In Lo internally shorted, and the reference selected by the other control bits is used. The read zero conversion result is proportional to the internal offsets of the MAX134, and this result should be subtracted from other measurements to get zero-corrected readings.

Filter On and Filter Short: These bits control the active filter as follows:

Filter On	Filter Short	Function
1	0	Normal filter on condition
1	1	Filter on with Rfilter 1 bypassed
0	0	Bypasses the filter
0	0	Invalid combination

DC: This bit selects the DC mode when set to 1 and selects the AC mode when it is 0.

External AC: This bit should be set to 1 whenever the AC mode is selected (DC=0)

Divider Sense: This bit, the 10-0 through 10-4, and the Current bits select the input signal source. Divider sense should be 1 whenever the input attenuator is selected. Set Divider Sense to 0 to select the 400-mV input.

Current: Set divider sense to 0 and the Current bit to 1 to select the Current input. Note that while this bit and the associated pin are named "Current," the actual input is the voltage drop across an external current sensing resistor.

Table 1—The MAX134 has a host of control bits that allow complete processor control.

Address or Register Number	Register Name	Register Contents
0	Ones	Conversion Result BCD data for least-significant digit (the undisplayed digit used for digital autozero)
1	Tens	BCD Data of Conversion Result (least-significant displayed digit)
2	Hundreds	BCD Data of Conversion Result
3	Thousands	BCD Data of Conversion Result
4	10 Thousands	BCD Data of Conversion Result
5	status	D3: Always 1 D2: Latched Continuity D1: Holding D0: Low Battery

Table 2—the register map of the data out from the MAX134 to the processor shows the use of BCD data for the conversion results.

read on a digital meter, and set the pots for an output value of 2.00 VDC.

A SINGLE-CHANNEL DVM INTERFACE

Figure 1 is the minimum circuit configuration for attaching a single-

channel MAX134-based DVM interface to an 8031/8052 processor. To save wiring time I built the prototype on a BCC55 prototyping card generally used with the BCC52 controller. The signals required to interface to the MAX134 are very straightforward and it should work with most processors.

The entire circuit is designed to operate on a single +5-V supply. The MAX134 (and the LM324 in the AC-to-DC converter) operates on +5 V and -5 V. The negative voltage is derived from an ICL7660 charge pump inverter. The ADC reference also is unique. Unlike many chips that require 5-V references that run from higher voltages, the MAX134 has a very low reference. Using an ICL8069 1.2-V reference chip and a divider network, we choose either a 545-mV or 655-mV reference. The difference in the voltage changes the integration time. For those of us with a lot of 60 Hz around, we should use 545 mV.

Other than that, just a few gates are necessary to implement the bus interface. The MAX134 uses its own 32-kHz crystal and operates independently of any computer timing. To set the registers we merely write to addresses CA00–CA04 hex with the values selected from Table 1. The input registers are double-buffered and writing to them does not immediately affect operation until after EOC (End

Tired of waiting for the prompt?


DOS IN ROM!

Hard drives may take even longer to boot than floppies because of complex autoexec files. End the wait with an MVS ROM DRIVE !!!

Works like Tandy or Toshiba with YOUR software instead of theirs. Perfect for hostile environments or highly secure diskless setup.

- BOOT XT/AT IN LESS THAN 0.1 SEC
- WORKS WITH ANY DOS, ANY PROGRAM
- FIVE YEAR LIMITED WARRANTY
- EXTEND LIFE OF MECHANICAL DRIVE
- WORKSTATIONS, CONTROLLERS, LANS

MVDISK1 64k	\$150
MVDISK2 360k	200
MVDISK3 720k	300
MVPRGM programmer	95
WEDGE ter utility	50



WORLD'S SMALLEST PC !!!

ROBOTS ALARMS RECORDERS DOS

8088 SINGLE BOARD COMPUTER

- DEVELOP AND DEBUG ON XT/AT
- DOWNLOAD TO SBC TO TEST
- PROGRAM EPROM, STAND ALONE
- BATTERY OR 5 VDC OPERATION
- FIVE YEAR LIMITED WARRANTY
- 2 PARALLEL, 3 SERIAL, TIMER
- PC TYPE BUS, 5/10/20 MHZ

MVSBC1 cpu,io,ram ..	\$150
MVBIOS rom+pc disk...	50
IBMCBL pc cable.....	40
IBMPRT pc interface...	75
RAM32K cmos memory...	60

FREE SHIPPING !!

Send chk/case for info to:

MERRIMACK VALLEY SYSTEMS
Box 994 Merrimack, NH 03054
Phone (508) 792-9507



Reader Service #157

ROM-IT

EPROM EMULATION SYSTEM



- Emulates up to 8 1-Megabit EPROMS with one control card.
- Downloads 1-Megabit programs in less than 10 seconds.
- *Accepts Intel Hex, Motorola S-Record, and Binary files.
- Software available for IBM PC and Macintosh systems.
- *Allows examination and modification of individual bytes or blocks.

Call or fax today for more information!

Base 27256 EPROM System \$395.00
Other configurations available.



Incredible Technologies, Inc.
709 West Algonquin Road
Arlington Heights Illinois 60005
(708) 437-2433 Fax (708) 437-2473

Visa, Mastercard, and American Express accepted

Reader Service #143

8031 μ Controller Modules

NEW!!!

Control-R II

- ✓ Industry Standard 8-bit 8031 CPU
- ✓ 128 bytes RAM / 8 K of EPROM
- ✓ Socket for 8 Kbytes of Static RAM
- ✓ 11.0592 MHz Operation
- ✓ 14/16 bits of parallel I/O plus access to address, data and control signals on standard headers.
- ✓ MAX232 Serial I/O (optional)
- ✓ +5 volt single supply operation
- ✓ Compact 3.50" x 4.5" size
- ✓ Assembled & Tested, not a kit

\$64.95 each

Control-R I

- ✓ Industry Standard 8-bit 8031 CPU
- ✓ 128 bytes RAM / 8K EPROM
- ✓ 11.0592 MHz Operation
- ✓ 14/16 bits of parallel I/O
- ✓ MAX232 Serial I/O (optional)
- ✓ +5 volt single supply operation
- ✓ Compact 2.75" x 4.00" size
- ✓ Assembled & Tested, not a kit

\$39.95 each

Options:

- MAX232 I.C. (\$6.95ea.)
- 6264 8K SRAM (\$10.00ea.)

Development Software:

- PseudoSam 5.1 Software (\$50.00)
Level II MSDOS cross-assembler.
Assemble 8031 code with a PC.
- PseudoMax 5.1 Software (\$100.00)
MSDOS cross-simulator. Test and debug 8031 code on your PC!

Ordering Information:

Check or Money Orders accepted. All orders add \$3.00 S&H in Continental U.S. or \$6.00 for Alaska, Hawaii and Canada. Illinois residents must add 6.25% tax.

Cottage Resources Corporation
Suite 3-672, 1405 Stevenson Drive
Springfield, Illinois 62703
(217) 529-7679

Reader Service E127

of Conversion). EOC can be connected to an input line or an interrupt input.

After EOC goes high, the $\pm 40,000$ -count ADC value is obtained by reading five registers at locations CA00-CA04 hex. A sixth register at CA05 hex contains status information. The data format is nine's complement BCD (say what?). BCD readings of 00001, 00100, and 40000 represent measurement values of +00001, +00100, and +40000, respectively. BCD readings of 60000, 99900, and 99999 represent values of -40000, -00100, and -00001, respectively.

This may seem strange but for a computer it is an easy matter to convert it to something more understandable. After taking a reading, just test it for a range of 60000-99999 or 00000-40000. If the former, then subtract it from 10,000 and add a negative sign. If the latter, then take the absolute number with a plus sign.

SOFTWARE CALIBRATION

Since the MAX134 was designed to be attached to a microprocessor you can take some license in the hardware design and "fix" it in software.

First you would perform a "read zero" by setting the internal registers to short the ADC's IN HI and IN LO connections. This zero reading is then an offset correction which is added to or subtracted from any actual signal readings. **Reading zero** should be performed periodically to maintain proper calibration.

Once zeroed, next you apply a known input voltage for each range and note the difference in the reading obtained versus the actual applied value. The difference between measured and actual then becomes a scaling factor by which all subsequent measurements can be multiplied to obtain the proper displayed value.

Software can also be taken a couple steps further. Besides scaling and correcting readings, software can perform autoranging. **Though various techniques exist, the basic premise is to set the lowest input range and then increase ranges until the applied input is not out of the selected range.** Autoranging does however take ad-

ditional conversion time and is most useful on single-channel applications.

ADDING MORE CHANNELS

Because the MAX134 chip does so much, little is required to make a single-channel DVM interface. Unfortunately, real-time control in an embedded control application, like the hydroponic gardening I described, requires more than one ADC input.

The easiest and least expensive way to increase the number of channels on an ADC is to add an input multiplexer. This is most often accomplished by adding a CMOS multiplexer chip such as the MUX08.

Unfortunately, CMOS multiplexers will not completely work in this application and could present a safety hazard. Simply put, they won't hack the voltages. Most muxes are **good** for +15-V inputs (using 418-V supplies) and while there are even a few new ones on the market that won't flash into incineration till ± 50 V, none are rated for the full input range of this interface. The only economical switching device which can withstand 400 V safely is a relay. The added advantage of using relays is that if we specify DPST (double-pole single-throw) relays, the interface cannot only be multichannel but differential input (more later).

Figure 2 outlines the circuit to add eight DPST relays to the single-channel DVM interface. It merely consists of a 74LS259 addressable latch and a relay driver chip. To set channel #4 we merely write a value of 04H to address C900H. To set #2 as the next channel, we send 00H first to turn off the other relay and then 02H to set #2. Of course, you should allow about 20 milliseconds for the relays to operate, but if you are using BASIC to run the interface (quite conceivable since the MAX134 only does 20 conversions per second), critical timing is a moot point.

One last comment on the relays: I originally said that this was a +5V-only interface. It still is if you use 5-volt relays. In my prototype I chose to use 12-volt relays simply because I had them and 12 volts was available on the BCC bus.

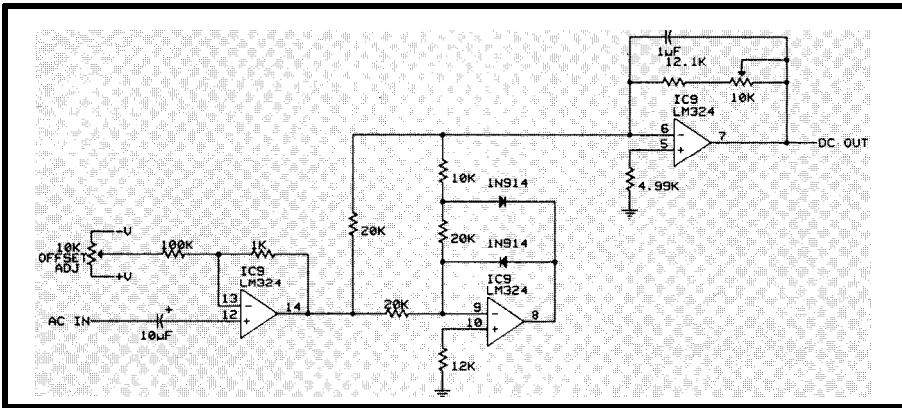


Figure 4—The AC-to-DC converter used in the interface is a typical half-wave design.

As usual, I also added some extra-aneous circuitry that good programmers can omit. EOC can be handled either through an interrupt line or a parallel input port. To make sure EOC was working properly while not having to mess with interrupt handling routines, I took the parallel route. Since I had seven bits left, I added a 4-bit "user input" DIP switch so that I could write multiple test routines and switch-select them when operating the interface without a terminal. This is

just frosting and not required to run the interface.

MULTICHANNEL DIFFERENTIAL INPUT

This DVM interface gains most of its performance from the microprocessor and program running it. As block diagrammed in Figure 5a, you could use the circuit from Figure 1 on a prototyping card and add a wide-range ADC to any processor bus. It can even be expanded to multichan-

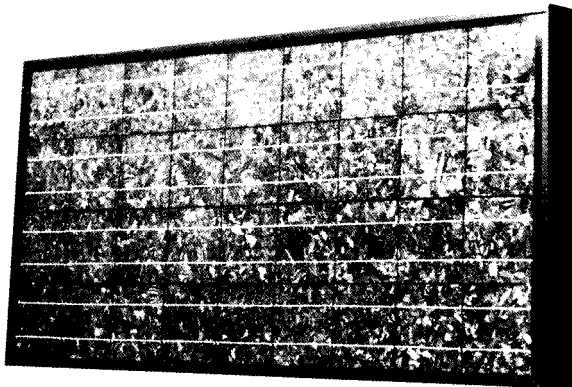
nel AC-to-DC operation as my final prototype demonstrates.

We have to be careful how we use a multichannel ADC with such a wide range. Since we use isolated battery-operated DVMs to read high voltages with ease, we must not infer the same safe use with unisolated ADCs.

While the single-channel DVM interface connects to the applied input through two wires, in actuality it has what is called a "single-ended" input. One wire goes to the ADC input and the other wire goes to the combined analog and digital ground. In the typical embedded controller, this ground is carried through to the power supply and the signal ground (pin 7) on the RS-232 cable. In some cases it may also be earth ground as well.

If you connect the DVM interface inputs to the 120-VAC or 220-VAC line to read it (an acceptable range for our board), you may or may not be in trouble. If the hot side of the line (black wire) goes to the ADC's 10M input and the return side (white wire) is connected to ground you will read

ELECTRICITY from SUNLIGHT



Photovoltaics for remote and online power applications. Reliable, renewable electricity from sunlight. Where there is a battery to be charged, there is a place for photovoltaics.

SUNNYSIDE SOLAR

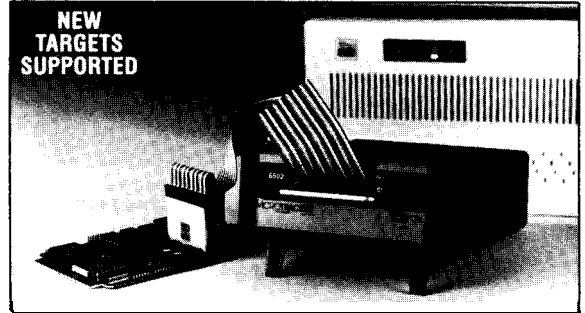
RD4 Box 808 Green River Road
Brattleboro, Vermont 05301

802-257-1482

Or circle 189 on the Reader Service Card

MICROTRACKER™

Real-Time Software Analyzer/Debugger



The MicroTracker™ can significantly reduce the cost of your next real-time product development project. Advanced features speed software development and enhance quality assurance.

FEATURES

- 2K or 8K Trace Memory
 - Interval Timer
 - Performance Analysis
 - RS-232 Interface
 - IBM PC Software
 - Symbolic Disassembly
 - Low Cost from \$1295.
 - Instruction Disassembly for Z80, 8085, 6502, 6802, 6809, 8031/8051, 80188/80186, V40/V50
- Call for Free Brochure

LOGICAL
ADVANCES

52 W. HENDERSON RD.,
COLUMBUS, OHIO 43214
(614) 267-4405

everything fine. If, on the other hand, your house wiring has been reversed, then you could have a direct short between the ground side of the DVM interface and the AC line. The correct alternative is to isolate the complete DVM interface the same way the battery-operated DVM is.

Figure 5b diagrams the elements of an isolated multichannel DVM interface. Using the same singleboard microcontroller and DVM interface electronics as before we merely add an isolated power supply and RS-232 transceiver.

The optoisolated RS-232 transceiver is shown in Figure 6 "steals" power from the connecting RS-232 source. To function, the source has to have "true" RS-232 voltage levels of ± 9 V or greater without a half mile of wire. While I can't guarantee 100% operation under all conditions, it has never failed to work for me.

The power supply can be any modular plug-in unit whose output

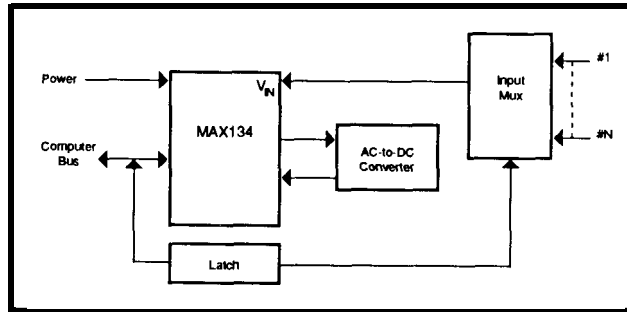


Figure 5a—The minimum-configuration multichannel interface can be used with most any processor bus.

side ground lead is not connected to earth ground or the power line return. This sounds like a complicated requirement, but 99% of the modular supplies I have tested or used are made this way. Just realize that the isolation rating of the system becomes that of its weakest part. If you use a wall module with 500-volt isolation and optoisolators in the RS-232 converter with 2500-volt ratings, the DVM interface will still only have 500-V isolation. Beware of measuring 1000-V inputs in this configuration.

If you want 2500-volt isolation then you either have to find a better-

rated power supply or operate the unit on batteries.

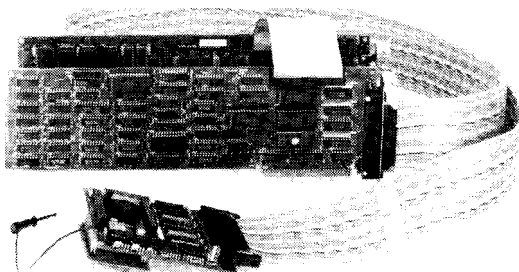
Once you understand isolation then we can talk about differential input. Basically, it is just another way to say that the measured value is the voltage between two points relative to each other. The DVM interface has two wires which measure voltage but, as I just explained, unless

the DVM is isolated you can't necessarily reverse the leads. By isolating the complete DVM interface, RS-232, and power supply, the two inputs then become "floating differential inputs" with respect to anything they measure. You can put them across the AC line, a battery, or in the signal path of any circuit, and they will only register the voltage between the two probes.

The function of the eight relays added for multichannel operation is also consistent with this capability. Only one relay is energized at a time and both input lines are switched together. Because we disconnect the

68HC11

PC-based emulator for 68HC11



SEE EEM 89/90
Pages D 1324-1326

- PC plug-in or RS-232 box.
- Pull-down menus with full window support, combined with command-driven User Interface.
- Up to 16 MHz real time emulation.
- No intrusions to the 68HC11's resources.
- 48 bit wide 16K deep trace. All functions usable without disturbing emulation. Time stamping. Two level trigger.
- Symbolic and C Source Level Debugging, including in-line assembler and disassembler.
- Supports A, E, D and F parts.

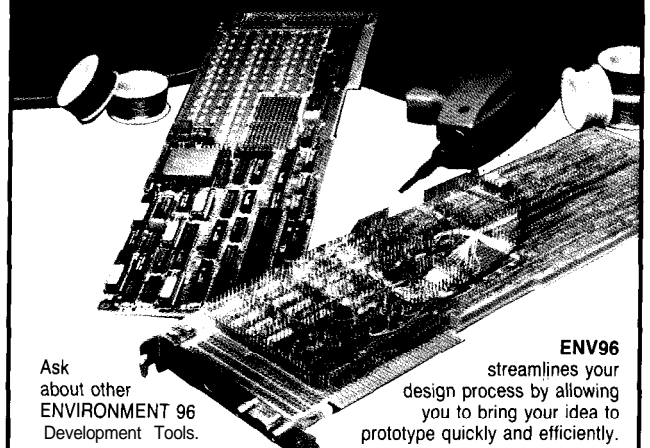
Prices: 64K Emulator and pod \$2590; 4K Trace \$1995 'US only

CALL OR WRITE FOR FREE DEMO DISK!

NOHAU CORPORATION
51 E. Campbell Avenue
Campbell, CA 95008
(406) 866-1820 FAX (408) 378-7869

SEE US AT WESCON BOOTH #1762

COMPLETE 8096 DEVELOPMENT ENVIRONMENT



Ask about other ENVIRONMENT 96 Development Tools.

ENV96 streamlines your design process by allowing you to bring your idea to prototype quickly and efficiently.

ENV96 is a plug-in development board for Intel's 8096/80C196 microcontroller. ENV96 includes an 8097BH or 80C197KB, processor support hardware, a shared 64K RAM module, an interface to the PC, a wirewrap area with digital and analog power and ground distributed throughout, and wirewrap pins on the PC Bus and processor pins. Easy to learn Debug software simplifies HW/SW debugging.



Annapolis Micro Systems, Inc.

190 Admiral Cochrane Drive Annapolis, MD 21401
(301) 841-2514 FAX: (301) 841-2518

presently addressed channel before energizing another, the eight inputs can actually be looked at as eight pairs of differential inputs which don't interfere with each other.

As presented, each channel can be set to either AC or DC input and an entirely different range. All you do is poke in the register selections and record the results. Being independent channels and differential inputs also offers one other measurement potential. So far we have only spoken of voltage measurement, but current measurement is nothing more than measuring the voltage across a known shunt resistor. Using the 400-mV scale with a 0.01-Q shunt across one of the relay inputs turns that channel into a 8-4-A ammeter input (AC or DC). Finally, while I haven't tried it, it's also conceivable that you could measure resistances by reading the voltage across an unknown resistance with a known current applied. That's how a DMM does it so why not here too.

INCONCLUSION

The real power of this DVM interface is the computer controlling it. It has the basic range switching and AC-to-DC conversion capabilities built in relatively little hardware. What turns this unit into a truly functional process-control interface is software. Autozeroing, calibration, and scaling are all software functions. I don't claim to be a programmer, and I'm like that teacher you used to have who always claimed that the hard stuff was "left as an exercise for the student." I've posted the BASIC-52 listing of my initial DVM interface test program on the Circuit Cellar BBS as an illustration. **[Editor's Note: Software for this article is available for downloading from the Circuit Cellar BBS or on Software On Disk #17. For downloading and purchasing information, see page 208.1]**

Now that I have an ADC interface suitable for my hydroponic experiment, I'll get back on it. Coincidentally, a shipment of 24 solenoid valves (just in case) arrived today. I must state that I am not promising to make such a garden; only to see if it is possible first. My next step is to set up a

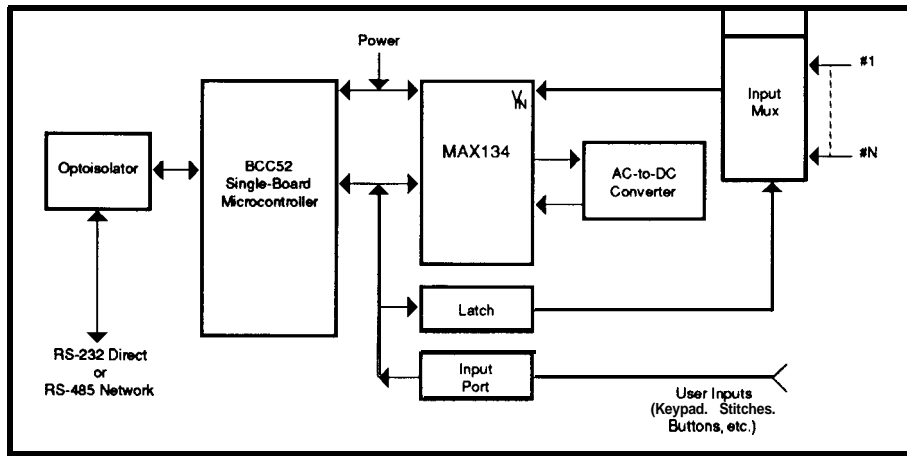


Figure 5b—An isolated power supply and isolated RS-232 interface added to the base circuit combine to make up a completely isolated DVM interface.

Cross-Assemblers from \$50.00
Simulators from \$100.00
Cross-Disassemblers from \$100.00
Developer Packages
 from \$200.00 (a \$50.00 Savings)

Make Programming Easy

Our Macro Cross-assemblers are easy to use. With powerful conditional assembly and unlimited include files.

Get It Debugged--FAST

Don't wait until the hardware is finished. Debug your software with our Simulators.

Recover Lost Source!

Our line of disassemblers can help you re-create the original assembly language source.

Thousands Of Satisfied Customers Worldwide

PseudoCorp has been providing quality solutions for microprocessor problems since 1985.

Processors

Intel 8048	RCA 1802,05	Intel 8051	Intel 8096,196kc
Motorola 6800	Motorola 6801	Motorola 68HC11	Motorola 6805
Hitachi 6301	Motorola 6809	MOS Tech 6502	WDC 65C02
Rockwell 65C02	Intel 8080,85	Zilog Z80	NSC 800
Hitachi HD64180	Mot. 68k,8,10		

New

Zilog Z8

Zilog Super 8

- All products require an IBM PC or compatible.

For Information Or To Order Call:

PseudoCorp

716 Thimble Shoals Blvd, Suite E
 Newport News, VA 23606

(804) 873-1947

FAX:(804)873-2154

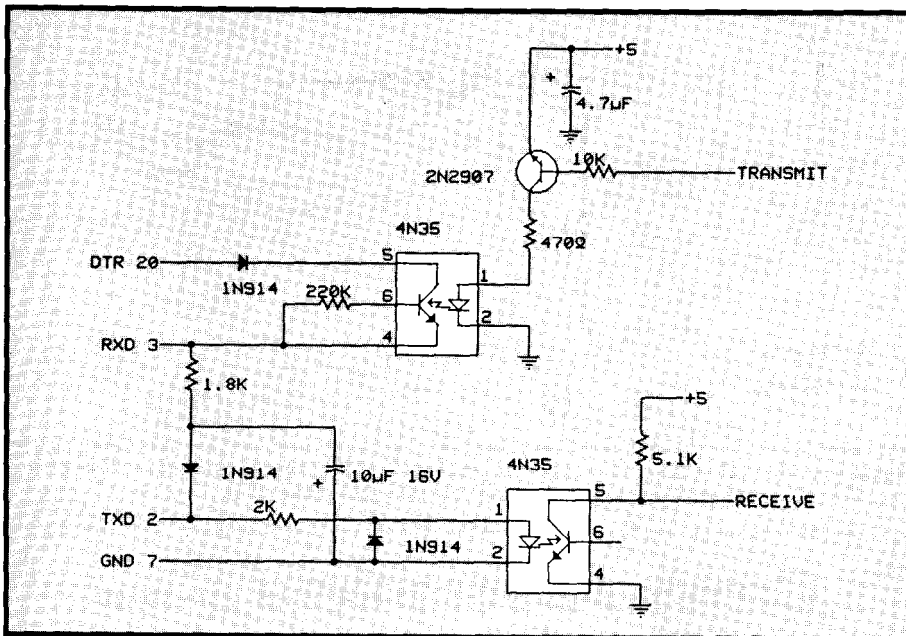


Figure 6—The optoisolated RS-232 transceiver "steals" power from the connecting RS-232 source.

little automatic plumbing system over the winter and see if I can automatically mix and measure chemicals. If so, then the next step is a real hydroponic garden. If you have any experience or good materials that might help, please let me know.

The greatest hurdle in accomplishing my experimental goal may be trying to get my greenhouse back. My wife has become quite attached to it and I'll have to contend with displacing 2000-3000 plants next spring. Filling up the solarium with either tubes

and chemicals or thousands of green leafy critters is out of the question. I spent last spring stepping over flower boxes while the greenhouse was being built. Unfortunately, the solution is all too obvious if I get to the implementation stage. I have this awful feeling that the only way I'll get any greenhouse room is "his" and "hers" greenhouses. I'll keep you posted. +

The MAX134 digital multimeter chip and data sheet are available from

circuit Cellar Kits
4 Park St., Suite 12
Vernon, CT 06066
(203) 875-2751

for \$35 postpaid (add \$5 outside U.S.).

Steve Ciarcia (pronounced See-AR-see-ah") is an electronics engineer and computer consultant with experience in process control, digital design and product development.

IRS

259 Very Useful
260 Moderately Useful
261 Not Useful

COMPLETE OPERATING SYSTEM UNDER \$20



HOW can I get this done?

Use the F68HC11!



Well, what I really want is a CMOS computer system for dedicated applications, that has low enough power requirements to be solar-powered if need be, with WAIT and STOP modes to really cut down on power consumption when necessary.

It's got to have some advanced features, too, like a built-in, high-level language and an operating system that can autostart my user applications without a lot of hassle.

It should have some built-in EEPROM and some scratch pad RAM.

Boy, for those imbedded applications, it's got to have a watchdog timer system that checks for the computer operating properly and resets the system if there's a power glitch or something. Let's see, for I/O I usually need several parallel ports and perhaps a serial port or two.

and a 16-bit timer system that can handle some inputs to latch the count and some outputs that can be set up to toggle at the correct time without further processor attention and maybe a pulse accumulator.

An A/D converter, with a couple channels would sure be the ticket! It would have to be fairly fast, though, and maybe be taking readings all the time, so the processor can just get fresh data when needed.

And maybe there's a way I could do my editing on a PC and download the source to the dedicated system. Perhaps it could even put the downloaded program into its own EEPROM.

But really, the final system requires a low dollar unit, it just can't cost too much. It would be nice if it were smaller than a bread basket.

I wonder how much the first prototype is going to cost this time? It sure would help if there were a pretested, full up version of the system, with a prototyping area built on, and maybe even a target version of that same system.

Yeah, I may be dreaming, but if one existed, I'd buy it in a minute.

Hey! I operate on 10ma typical at 8 Mhz, lower in WAIT mode, with a STOP mode in the 2µa range.

I've got a full featured FORTH and an operating system that can easily autostart an internal or external user program.

How 'bout 1/2K EEPROM and 1/4K of RAM.

My watch dog timer and computer operating property circuit is built-in and programmable.

Configure me with 5 S-bit parallel ports, or 3 with a 64K address and data bus.

I've got two serial ports, one that's async and one that's sync.

My 16-bit timer has three input captures and 5 output compares and is cascadable with my S-bit pulse accumulator.

You want A/D? How 'bout S-bit, 8 channels, ratiometric, 16µS conversions, with continuous conversions possible on four selected channels.

I've been known to carry on a conversation with communication packages and I've got built-in EEPROM handlers.

How about \$37.25 in singles? **under \$20** in 1000 piece volume?

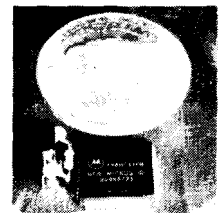
You can hide a complete operating system under a \$20 gold piece.

Listen, you can buy the NMIX-0022PS full development system for \$290.

(Try getting a hoard wire wrapped for that price).

The NMIT-0021 target board is \$90.

Hey, I'm available for immediate delivery!



Manuals only \$30 items in limited stock.

68HC11 hardware by Motorola, Inc., F68HC11 Max-FORTH™ internal firmware by New Micros, Inc., NMIX and NMIT series boards by New Micros, Inc.



NEW MICROS, INC.
1601 CHALK HILL ROAD
DALLAS, TEXAS 75212
214/339-2204

FEATURE ARTICLE

Thomas Mosteller

Control Theory for Embedded Controllers

An Introduction to the Basics of Computerized Control

The design of a successful embedded control system requires knowledge of several different disciplines. Once you have your microprocessor hardware up and running, you have to write controlling software (or find someone to do it for you).

How will that software work? "Simple," you think, "I'll read in the variable I'm trying to control, and if it's too low I'll crank the output up a little, and if it's too high I'll turn it down some. With a little fiddling, that ought to do the job."

In some cases, that approach will work. However, a lot of times it will lead to hours of frustration and attendant cursing. Even worse, you may arrive at an ad hoc solution which will stop working when you turn your back.

These problems can be circumvented with a little knowledge of control theory. This is a discussion which skips most of the math, and especially all of the heavy stuff, with an emphasis on real-world problem solving techniques.

I'll start with an example system. Suppose you build a hot tub on your deck and you want to install a heater to keep the water at a constant 110°F. Being an avid **CIRCUIT CELLAR INK**

reader, you decide to bypass all the commercial systems and use an embedded controller to perform this critical task.

Conceptually, the system will look like the one shown in Figure 1. A temperature sensor which we've dunked in the water is connected to the controller. Here a suitable A/D

AN ON/OFF CONTROL SYSTEM

The simplest kind of control system uses on/off control, which can be summed up: If the water's too cold, turn the heater on; if it's too hot, turn it off. The type of results attainable with this kind of system are best illustrated by the familiar home heating system, where a variation of two or three degrees Fahrenheit can be achieved. On/off control is especially appropriate when it's not convenient to modulate the output of the end element you're using, such as gas furnaces or air conditioning compressors. In our system, we could use a relay to control the heater.

This control scheme is easily implemented in software. Using the

hot tub system mentioned above as an example, you would begin by reading the water temperature in through the A/D converter and storing it in a variable called *Water_Temperature*. The desired value of the ambient temperature is stored in *Set_Point*. A simple logic statement of the form:

```
IF (Water_Temperature - Set_point) > 0 THEN  
    Heater = OFF  
ELSE  
    Heater = ON
```

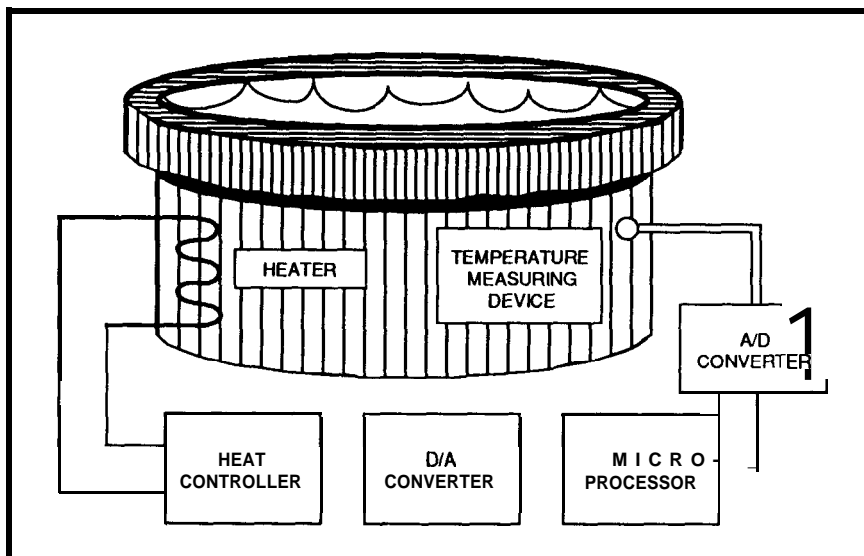


Figure 1 - A simple closed-loop embedded controller application might control the water temperature of a hot tub.

conversion method is used, and the result of this conversion is available as input to a microprocessor. The microprocessor then manipulates this data, and controls an output port which in turn controls the heater. All in all, a practical system except for the minor oversight that we haven't provided a means of isolating the electrical system from the inhabitants of the hot tub to prevent electrical shock. As they like to say on TV, don't try this at home.

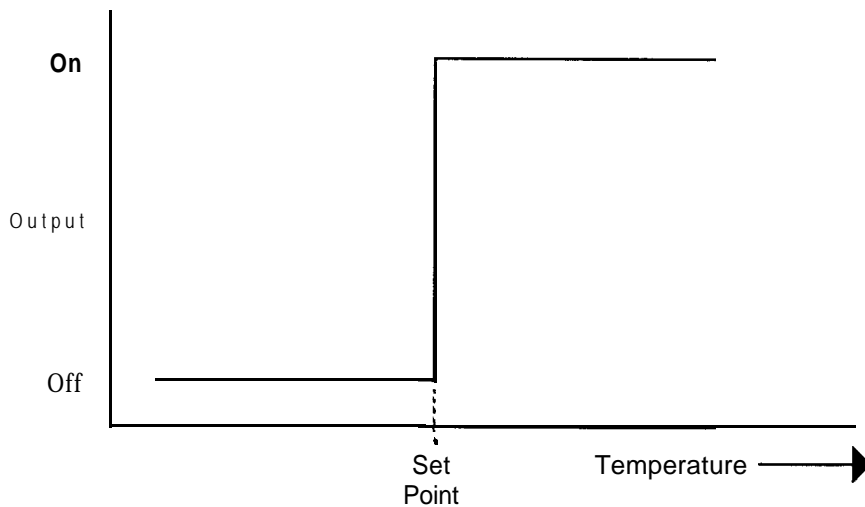


Figure 2a—Simple on/off control changes states anytime the *setpoint* is crossed.

would store the proper result in the variable called Heater. After that, it's a simple matter to test the value of Heater and use your I/O circuitry to cause the heater to turn on or off appropriately. The transfer function for this scheme is shown in Figure 2a.

One thing to watch for with this simple scheme is noise on the input signal. When the water temperature is close to the set point, a few counts of noise from the ADC will cause the end element to turn on and off rapidly or " chatter." Some end elements, like the heater in our example, are not affected by this, but it can cause unnecessary wear on devices like the controlling relay. In particular, compressors can be destroyed by rapid on/off cycling, and require built-in protection to prevent this.

Input noise can be addressed several ways. One technique is to low-pass filter the input signal to remove the bulk of the noise. This works, but has the side effect of delaying the system's response to rapidly changing inputs, which can sometimes be a problem. However, even with low-pass filtering, a count or two of uncertainty in the measured variable is hard to avoid.

A better solution is to add positive feedback, or hysteresis, to the system to establish a "guard band" around the set point where the output won't change (as shown in Figure 2b). This is accomplished in our example by changing the line above to:

```
IF (Water_Temperature - Set_Point) > Guard_Band THEN
    Heater = OFF
OR
IF (Water_Temperature - Set_Point) < -Guard_Band THEN
    Heater = ON
```

The statement above can be thought of in the following way: Assume the guard band is 1". If the water temperature is within one degree of the set point, either above or below, that's close enough; don't change the heater's state. If it goes beyond those limits, then turn the heater on or off appropriately. As a general rule of thumb, if you need more than a few percent of hysteresis to keep your output stable, then you've got noise problems which should be properly dealt with.

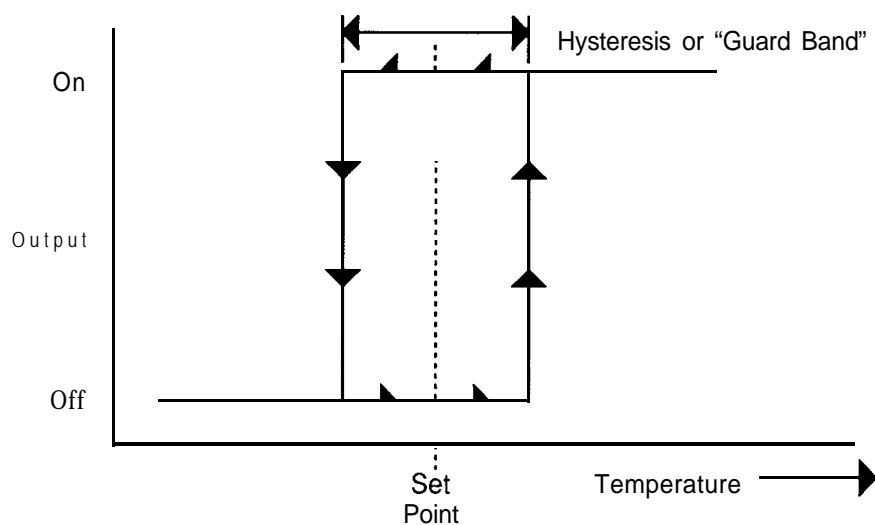


Figure 2b—On/off control with hysteresis adds a guard band around the setpoint.

On/off control should be sufficient to keep our example system's water temperature within three or four degrees of the desired set point. In most cases, that should be good enough. But suppose it isn't?

PROPORTIONAL CONTROL

Proportional control is the simplest control scheme which allows the end element controlling the process to assume a continuous spectrum of intermediate states (well, almost; the output of the analog to digital conversion process will always impose a finite limit on resolution). This can result in much "finer" control with a smaller deviation of the process from the intended set point.

Proportional control is accomplished in software by a statement of the form:

$$\text{Output} = (\text{Set_Point} - \text{Water_Temperature}) \times \text{Gain} + \text{Constant}$$

This equation has a couple of interesting terms. The first one is the gain, which expresses how quickly the output changes for a given input change. Note that as gain is made very high, you essentially end up with on/off control again, as a very small change in the input will cause the output to swing from fully off to fully on. In practice, the gain used is a compromise between accuracy (high gain) and stability (low gain).

The second term the designer must determine is the constant offset. This constant is required because some end element output is usually required when the set point and process variable are equal. While it's possible to calculate the value of the constant, we'll empirically determine this constant by manually tweaking the heater's output until the water temperature settles at the desired set point. This is a lot easier, and in most cases the results are more accurate. However, if you're trying to control the processing of a railroad car-sized load of expensive chemicals, "tweaking" may be frowned upon.

Readers who recall their high school algebra will recognize that the error is related to the output by the equation of a straight line: $y = mx + b$.

Implementing proportional control with our hot tub requires a major hardware change. While we previously controlled the heater with a relay, we now have to get a little more sophisticated and use a scheme capable of intermediate heater output

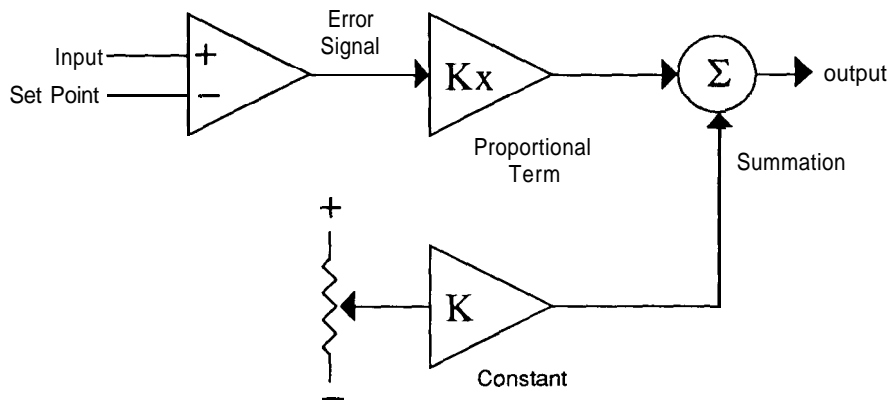


Figure 3a—A system which uses proportional control has much finer control than with simple on/off.

levels. In practice, this is usually apparent that there are a few problems with it. Since the output of the control algorithm is a specific constant when the error signal is zero, this kind of control scheme can only balance out under one unique set of circumstances.

PROPORTIONAL AND INTEGRAL CONTROL

If you think about the proportional control scheme, it becomes

In the case of our hot tub, suppose by trial and error we find that when it's 75°F outside, we can set the heater to 50% output, and the water will heat up to 110°F and remain there. We then

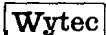
68HC11



WICE 68HC11: PC based real-time 68HC11 ICE

Easy to use, low cost & high performance 64K emulator. Real-time & full speed up to 14 MHz clock rates. No intrusion to target resources. Symbolic debugger supports many assemblers & C compilers including Motorola freeware. Windowed user-friendly interface. Data watch windows for memory, registers & stack. New concept in the ICE design, accesses host PC's keyboard and generates tones via PC's speaker from target program. User-defined windows on PC's CRT simulate any LED, LCD or VFD displays of target system in real-time (useful for new product demo, production testing or bench troubleshooting). Debugging software in real-time without target hardware. Logic analyzer trigger output, on-line assembler, disassembler, compare, dump, EEPROM, enter, fill, move, search, single step & breakpoint commands. 115.2K bps RS-232C link for fast upload/download. 30 day money back guarantee.

Complete system \$795 (708) 894-1440



Z8

Z8 emulator
w ICE Z8 \$995

Suite 140
18SC E. Lake Street
Bloomington, IL 60108

Announcing the

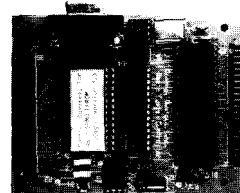
WB-1 80C31 Industrial Controller

and Peripherals

Our expandable controllers get your projects up and running *fast!*

Features include:

- ▶ Self-contained vertical stacking bus
- ▶ AT style DB9 RS232C serial port
- ▶ Built-in watchdog timer
- ▶ Built-in power fail detection
- ▶ 8K battery backed RAM
- ▶ 8K EPROM
- ▶ Small 4" X 3" board size
- ▶ All address and data pins available on 80 pin bus
- ▶ 16 line decoder for memory mapping
- ▶ Surface mount technology utilized for glue chips
- ▶ Voltage regulator (+5 VDC)
- ▶ Reset button
- ▶ High quality dry film solder mask pc board
- ▶ Industrial temperature range available
- ▶ Diskette of assembly language support software included
- ▶ 24 hour support bulletin board (918)251-8031



\$120⁰⁰

Stacking bus peripherals include:

- ▶ Breadboard with screw terminals and +5 VDC regulator WB-1 BRD
- ▶ LCD (20 X 4 char.). 16 button keypad interface & latched I/O WB-1 LCD
- ▶ Backplane with all power supplies and screw terminals for bus WB-1 BPS
- ▶ 82C55 parallel I/O board (total of 9 parallel ports) WB-1 PIO*
- ▶ Multichannel A/D and D/A board (12 bit resolution) WB-IAD*
- ▶ Opto-isolated 4 - 20 mA current loops with surge protection WB-1420*

* Denotes cards available in the near future.



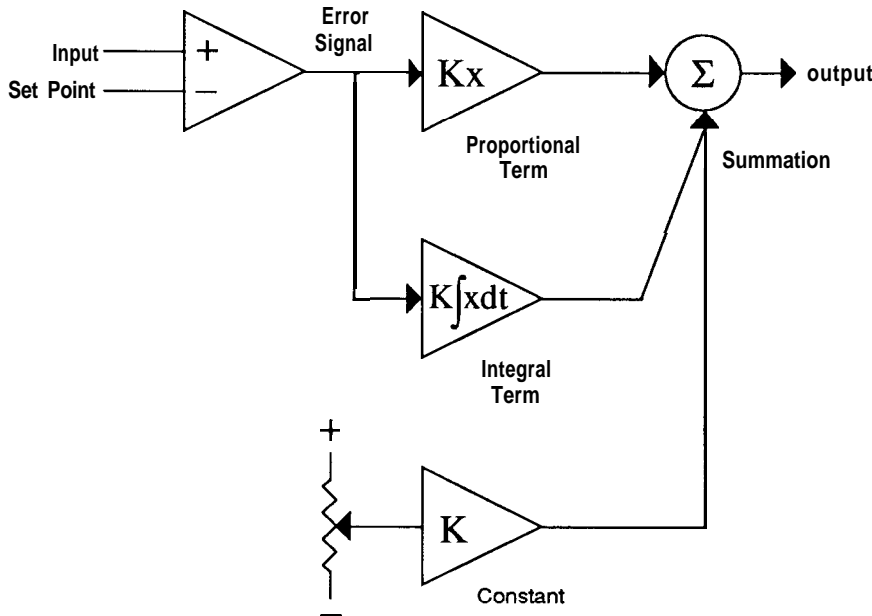
2009 West Detroit Street
Broken Arrow, Oklahoma 74012
(918) 258-6068 Voice
(918) 251-9851 FAX
(918) 251-8031 BBS

A Product Development Company

set our constant to 50%, and our set point to 110°F. For the rest of the day, we're happy. But now the sun goes down and the outside temperature drops to 50°F. Obviously, 50% output

is no longer going to be enough to keep the water at 110°F (recall that the output must be 50% when the water temperature is equal to the set point), and our tub is going to get colder.

How much colder depends on how high the gain is set, but the point is that the water temperature will "droop" away from the set point. Attempts to crank the gain way up to make the droop small will often result in system instability. What's needed here is a system with a constantly adjustable constant term.



This is exactly what integral control provides. By accumulating, or integrating, the error signal with respect to time, and feeding this term into the control algorithm, the process is forced to a zero error condition. In the process control industry, integral control was colloquially known as droop correction.

Now our control equation is getting a little more complicated. To simplify things, we'll break it down to a series of steps. The first thing to do is calculate the error signal:

$$\text{Error-Signal} = (\text{Process-Variable} - \text{Set_Point})$$

Figure 3b—Adding an integral term to the proportional control allows the system to adapt to changing conditions.

Now we must integrate, or keep a running sum of these error signals.

Total control with LMI FORTH™

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

For Development:

Interactive Forth-83 interpreter/Compilers for MS-DOS, OS/2, and the 80386

- 16-bit and 32-bit implementations
- Full screen editor and assembler
- Uses standard operating system files
- 500 page manual written in plain English
- Support for graphics, floating point, native code generation

For Applications: Forth-83 Metacompiler

- 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, 8086, 68000, 6502, 8051, 8096, 1802, 6303, 6809, 68HC11, 34010, V25, RTX-2000
- No license fee or royalty for compiled applications



Laboratory Microsystems Incorporated
Post Office Box 70430, Marina del Rey, CA 90295
Phone Credit Card Orders to: (213) 3067412
FAX: (213) 301-0761

Reader Service #151

THE TOTAL SOLUTION

FOR EMBEDDED DATA ACQUISITION AND CONTROL APPLICATIONS

EMAC OFFERS A COMPLETE LINE OF INDUSTRIAL STRENGTH SINGLE BOARD COMPUTERS AND SUPPORT PERIPHERALS, WITH ALL THE FEATURES NECESSARY TO PROVIDE YOU WITH A TOTAL SYSTEM SOLUTION! FEATURES INCLUDE:

- DIGITAL AND ANALOG I/O
- TIMER/COUNTERS
- RS232/422 SERIAL PORTS
- DISPLAY BOARDS
- TERMINAL BOARDS
- SIGNAL CONDITIONING CARDS
- EMBEDDED FORTH & BASIC LANGUAGES
- PRICES START AT \$249.00 QTY 1

EMAC, inc.

EQUIPMENT MONITOR AND CONTROL

618-529-4525

P.O. BOX 2042 CARBONDALE, IL 62902

Reader Service #132

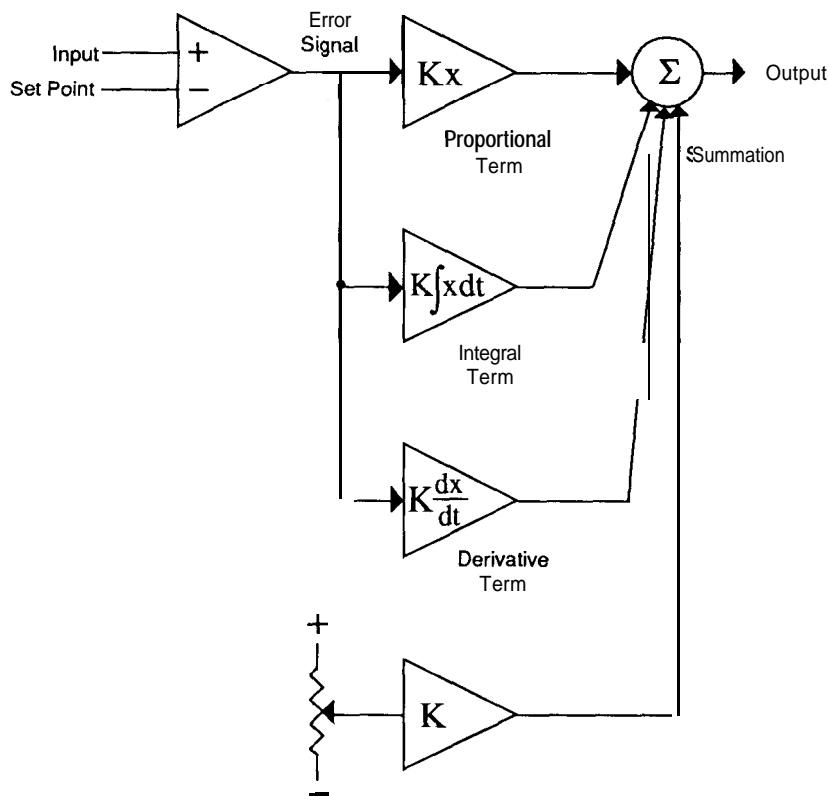


Figure 3c—By adding a derivative term to the proportional and integral terms, the system's response to sudden changes can be sped up.

Assuming we're presently at time N , we update a variable called *Integral_Term* by continuously adding the value of the error signal to the last value of the integral term:

$$\text{Integral_Term_Time_N} = \text{Integral_Term_Time_N-1} + \text{Error-Signal}$$

Finally, we'll combine these terms to get our output:

$$\text{Output} = \text{Gain1} \times \text{Error-Signal} + \text{Gain2} \times \text{Integral_Term}$$

Note that both the integral and proportional terms get their own independent gain terms, and that, in the steady state, no constant term is required—the integral term takes care of that. However, when the controller is initially turned on, the integral term will be zero, and quicker initial convergence can sometimes be managed by adding a constant.

Bear in mind that integral control is not an unmixed blessing. Consider what happens to our example system when it's turned on for the first time.

simple
multitasking
executive

smx
v2.0

for REAL-TIME EMBEDDED SYSTEMS

new features:

- Full MS-DOS compatibility
- 80x87 & math emulator support for reentrancy
- Stack size up to 64K per task
- Protected heaps & task-reentrant calls
- v25 register bank support
- PC demo source code
- Fast pipe macros

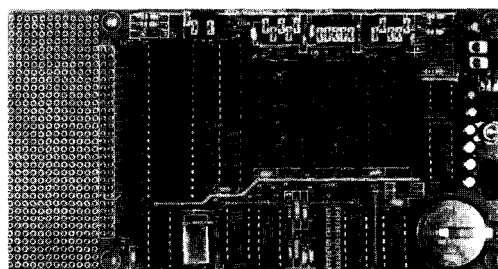
- 80x86 family
- ROM'able
- preemptive scheduler
- libraries for all memory models
- works with Microsoft% and Turbo[®]C, assembler and debugger
- 70 microsecond task switch (10MHz 80188)
- 15 microsecond interrupt latency
- 5KB to 25KB code size
- 1 year free support & updates
- 30 day money-back guarantee

Dev. Kit & No Royalty License \$1995
Source Code - \$1000
Evaluation Kit & PC demo - \$95
User's Guide & demo disk - \$25

MICRO DIGITAL

Cypress, CA 90630-5630
 1-800-366-2491
 FAX 714-895-2164

EMBEDDED CONTROLLER
for Quick Development



The EC-32™ is a versatile 80C32 microcontroller board. It is ideal for quickly developing products, prototypes or test fixtures.

- 80C32 microcontroller (8051 compatible)
- 35 mA operating current, 100 uA standby
- Program in C, BASIC or assembly language
- 8 to 92K RAM, EPROM or EEPROM
- Breadboard area and expansion bus
- RS-232 port and 12 digital I/O lines
- \$100

Iota Systems, Inc.

POB 8987
 InclineVillage, Nevada 89450
 (702) 831-6302

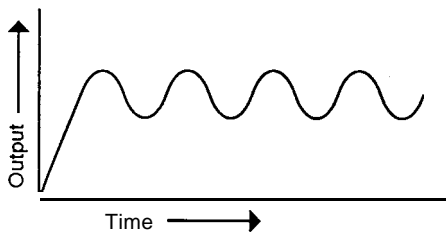


Figure 4a—A very high integral gain can cause sustained oscillation.

Of course, the water is well below the set point. This causes the integral error to begin growing rapidly. Since the heater can't bring the water up to temperature instantly, the integral error will continue to grow. In fact, a common error is to allow this number to get to be bigger than the microprocessor's register can handle, and thus wrap around from FFFF hex to 0000 hex, so be careful. However, even if a limit is imposed on the integral term, it can quickly get to be large enough to cause the output to go to 100% by itself.

Now think about what will happen as the water temperature nears the set point. The proportional error term will shrink, trying to cause the heater to cut back. However, the integral term has accumulated so much error in the time the water temperature has been below the set point that the heater output remains pegged at 100%. In fact, the output cannot begin to decrease until the water temperature goes above the set point so the integral term will begin to shrink. This will cause the water temperature to overshoot the desired point, and then come back down below it in (hopefully) an exponentially decaying oscillation.

Because of this phenomenon, a limit must be imposed on the amount of integral error which is accumulated. A simple way to do this is to test the output for being at its maximum or minimum limits. If it's at maximum, and the error signal is negative, don't update it. Why bother—you can't turn the heater up past 100% anyway! However, if the error signal is positive, do the update. The converse applies for the case where the output is at its minimum. These simple steps will

form an effective clamp and prevent—or at least limit—the oscillations mentioned above.

PROPORTIONAL, INTEGRAL, AND DERIVATIVE CONTROL

The last term we'll add to our control equation is the derivative term. Derivative control is used to speed up the system's response to sudden changes by feeding these changes directly to the output device, giving the system a "head start" on moving the output device in the proper direction to counteract the changes. Collo-

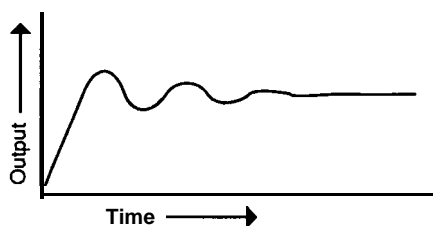


Figure 4b—Most systems work well with oscillations which die down after two or three cycles. Note the fast rise to the set point.

quially speaking, as the system runs away faster, we'll try even harder to catch up; this makes it harder for the process to diverge from a given set point.

This is accomplished in software by storing the last value of the error signal, and subtracting the new error signal from it:

$$\text{Derivative-Term-Time-N} = \text{Error_Term_Time_N} - \text{Error_Term_Time_N-1}$$

The value of the derivative term is then combined into the final form of our control equation:

$$\begin{aligned} \text{Output} = & \text{Gain1} \times \text{Error-Signal} \\ & + \text{Gain2} \times \text{Integral-Term} \\ & + \text{Gain3} \times \text{Derivative-Term} \end{aligned}$$

In some respects, the derivative control term is the most problematic of the three in the control equation. Since the gain of the derivative term goes up with increasing error signal frequency, any small change in the measured variable (especially noise) is amplified and passed to the end

PROGRAMS > 64K?

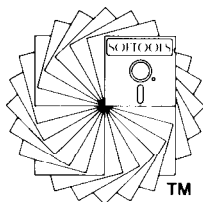
Auto-MMU Support Is The Answer.

**SASM-Advanced Macro Cross Assembler
SLINK-Advanced Linker**

Softools, Inc. introduces a relocating macro assembler and linker package that offers many features for the embedded programmer at an affordable price. It supports the 64180, Z80, 8085, and 2280 processors.

SASM also supports the 64180 MMU for automatic control of programs larger than 64K by making "long" calls into segments not mapped into the address space. It also includes many pseudo-opcodes for close compatibility with other assemblers. SASM accepts expressions that use operators common with other assemblers as well as C operator equivalents. SLINK is able to resolve any expression if SASM is unable to obtain a result. SASM includes a built-in MAKE facility which supports dependency file checks. It allows you to use one source file to generate a multi-module library file. In addition, SASM generates full source-level debugging information for each source file including the source name, include files, line numbers, public symbols, and local symbols.

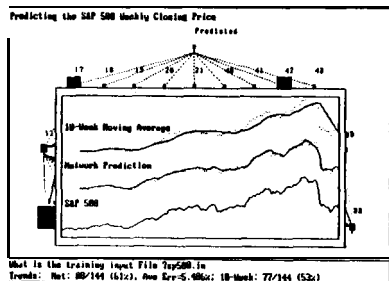
SLINK output is compatible with In-Circuit Emulator (ICE) source-level debugging, and also generates binary or Intel HEX files and has the ability to divide output into multiple ROM image files. It supports named segments which may be up to 64K in length each, and may be linked to reside at one physical address and executed at another. Any banked or MMU controlled program requires this feature to locate code effectively. SLINK also allows the exclusion of physical address ranges in order to leave holes in the output file.



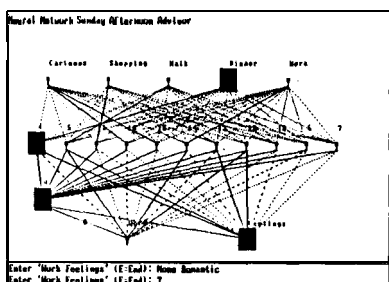
SOFTOOLS INC.
8770 Manahan Drive
Ellicott City, MD 21043
301-750-3733

ONLY \$249

Introducing NeuralWorks™ EXPLORER



Stock Market Forecasting



Expert Systems

NeuralWare, Inc. presents NeuralWorks Explorer, a neural network tutorial that provides the novice user with a method of learning neural network theory as well as an environment in which to build practical applications. Available on both the MAC and PC. Price \$199. VISA & MASTERCARD ACCEPTED

The NeuralWorks product line is currently used in:

- Oil Exploration
- Medical Diagnostics
- Industrial Inspection
- Credit Approval
- Process Control
- Insurance Underwriting
- Economic Modeling
- Noise Filtering
- Signal Processing
- Fraud Detection
- Bankruptcy Prediction
- Targeted Marketing



Penn Center West, Bldg. IV, Suite 227
Pittsburgh, Pennsylvania 15276

412-787-8222

Reader Service #166

element. This can make the output "twitchy" or "jittery." In our hot tub example, if someone were to spill a cold drink into the water near the temperature sensor, the derivative term could cause the heater output to suddenly go up to 100%, overheating the water.

There are a couple of ways to deal with this behavior. One is to keep the gain of the derivative term low, so that its contribution to the output is low. Alternatively, you can low-pass filter the derivative term to remove the twitchy nature. Since the derivative control term is itself a high-pass filter, it may seem strange to subsequently high-pass filter it! However, in practice, the frequency content of the measured variable usually allows the selection of filter constants which allow the system to pass most of the derivative term's useful information while suppressing useless noise.

TUNING THE SYSTEM

The equations above look pretty simple, and they are—with one exception: What are the values of the constants called out in the control equation? Determining this is called tuning the system, and it's the real skill in making a reliable, predictable system.

As a broad generalization, most systems work best when using proportional, integral, and derivative control. Some systems have special requirements which prevent the application of one of the types of control action. It can be difficult to judge in advance whether the system in question will support all three control modes. The best way to proceed is to tune the system per the following instructions. If the optimum gain for any of the control terms is less than 10% of the others, it can usually be omitted.

The basic principle of tuning a control loop is to set up the system in question while operating it in a manner which allows you to vary the tuning parameters. This can be done using an in-circuit emulator, or by making the application software smart enough to allow tweaking "on the

fly." The response of the system's measured variable to a step change is then observed on an oscilloscope or strip chart recorder. A step change is defined as a change in conditions made much more quickly than the system

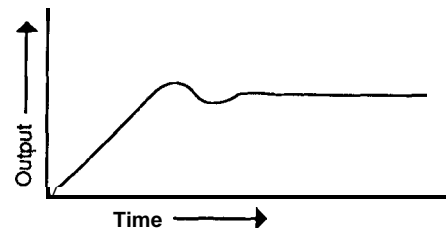


Figure 4c—Lowering the integral gain decreases the oscillations at the expense of slower rise to the set point.

can follow. This can be a tedious, time consuming process, so take notes of your results, as well as the tuning constants which produced them, to avoid having to repeat trials.

The nature of the system being tuned also has an effect on what will be considered optimum tuning. For example, in our hot tub, a small amount of temperature overshoot is acceptable in the interest of getting the water up to temperature quickly. However, this is not always the case. For instance, in a system used to add acid to a pH control system, overshoot would be intolerable: once the pH went past the set point, there would be no way to bring it back. Thorough knowledge of the system is required to ensure optimum results.

The first step is make sure that the response of the control equation is slower than that of the system being controlled. Otherwise, stability is

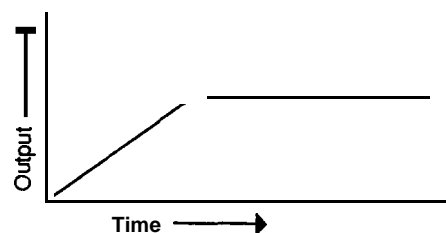


Figure 4d—Some critical systems require a highly damped response.

impossible. The importance of this statement cannot be overemphasized. Think about how you feel when people give you commands, and then change the commands faster than you can

execute them. Well, it drives control systems crazy too!

As an example, look at the physical arrangement of our hot tub temperature controller. Assume the embedded controller and controlling equation have a very fast response. However, because the water heater is on the bottom of the tub and the temperature measuring device is on the top, it takes a certain amount of time for the newly heated water to rise to the top of the tub and affect the measured temperature. In addition to this, the temperature measuring device has its own time constant, which adds on to that of the "water system." Together, these may account for a delay of 30 seconds.

Now think about what happens when the water temperature is below the set point. The heater heats the water, which raises the measured temperature reading. When the water temperature meets the set point, the controller lowers the heater's output (or cuts it off entirely, in the case of an

on/off control scheme). However, what the controller doesn't know is that due to the system's slow response, there's still more hot water on the way! Thus, the water temperature continues to rise. Perplexed by this event, the controller backs off even further until it's off completely. Then, the atmosphere cools the water off, but the controller doesn't do anything about it until the water temperature falls below the set point. Then, since there's a thirty-second delay for hot water to start appearing again, the controller will turn the heat all the way back up again. Now you've just invented the water temperature oscillator, which, although a noble accomplishment, is not exactly what you set out to do.

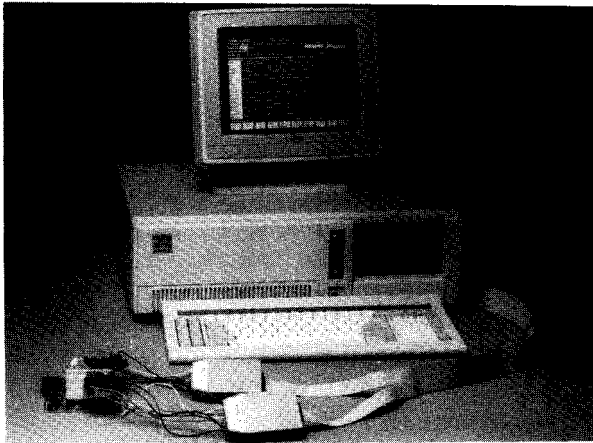
How do you get around this oscillation problem? You can either minimize the delays in the system, or slow the controller's response. For instance, moving the temperature sensor closer to the heater can dramatically cut delays, or you could install a mixer to

quickly distribute the heated water. Come to think of it, six or seven people splashing about in the tub would do nicely. As long as they kept moving, you'd have no problems.

However, the approach usually taken is to low-pass filter the controller's output to make it slower than the process. This is easily done, and it can be quickly changed to adapt to new conditions. The best way to determine the system's response speed is to manually turn the end element to a safe value, say 50% output. Wait for the measured variable to settle to its final value. Then, introduce a step change to the output, say to 70%. Time how long it takes to make 63% of the change in long-term measured values for 50% and 70% output, and that's the system's approximate time constant. The value used for the controller's time constant should be at least twice that, preferably five times for optimum stability.

Once the proper speed of the controller is established, the proper-

PC-Based Logic Analyzers



Sophisticated Logic Analysis a', Unsophisticated Prices

ID160 (50 MHz) for \$695

*ID161 (100 MHz) for \$895

• 50 MHz or 100 MHz Sampling • 8K Trace Buffer • 32-channel Operation
*Multi-Level Triggering *State Pass Counting *Event Timer/Counter
*Performance Histograms *Hardcopy Output *Disassembles popular 8-bit micros *and much more !

*30 Day Money Back Guarantee



INNOTECH DESIGN, INC.

6910 Oslo Circle, Suite 207
Buena Park, CA 90621

Tel: 714-522-1469 FAX: 714-527-1812

Reader Service #145

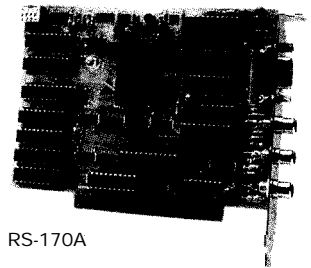
\$495 High Performance 1/2 the Cost

VIDEO FRAME GRABBER

- Real Time
- Hi-Res

CORTEX-I

- ✓ **Dual Resolution Modes**
Single Frame 512 X 484
Quad (4) Frames 256 X 242
- ✓ **256 Grey Levels** 8 Bits
- ✓ **Pixel Jitter** <1/10 pixel @ 512
- ✓ **Input LUT** Programmable
- ✓ **Software** includes Utility
"C" Library
Configurable as RAM Disk
DOS Prompt Commands
TIFF Conversion Utility
- ✓ **Composite Video** | B/W | NTSC RS-170A
RCA Phono / 9 Pin D-Sub
- ✓ External Trigger (TTL) with Prog. Delay
- ✓ Half Slot **IBM PC/XT/AT/386 Compatible**
- ✓ **90 Day Warranty** Parts and Labor
- ✓ **VISA/MC Accepted**



OEM/Dealer/Dist.
inquiries invited.

IMAGENATION CORPORATION
Specializing in Computer Vision

PO Box 84568, Vancouver, WA 98684-0568
Telephone/FAX (206) 944-9131

Reader Service #142

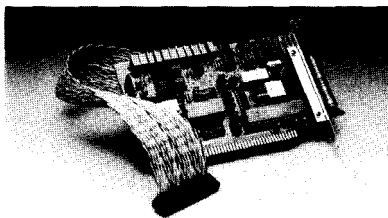
PC Bus Data Acquisition and Control

Quality U.S.-manufactured cards and software for single user, OEM, or embedded applications.

NEW PRODUCTS!

NEW CATALOG!

NEW PRICES!



ADA2100 - \$395

- 4 Diff./8 S.E. channels, 12-bit, 20 μ s ND
- Programmable gains of 1, 2, 4, 8, & 16
- Three 8-MHz timer/counters
- Two 12-bit D/A outputs
- 16 Digital I/O lines
- Dedicated ground for each analog signal!

Call today for our new catalog featuring the following hardware, software and much more!

AD1000 8 S.E. channels; 12-bit 20 μ s A/D; sample & hold; three 8-MHz timer/counters; 24 TTL digital I/O lines. \$275
 ADA1100 AD1000 with 2-channel D/A and resistor-configurable gain \$365
 AD2000 8 Diff./16 S.E. channels, 20 μ s A/D; sample & hold; three 8-MHz timer/counters; 2, 4, 8, 16 prog. gain; 16 digital I/O \$359
 ADA2000 AD2000 with 2-channel D/A and 40 digital I/O lines \$489
 AD100/AD500 Single-channel/S-channel, S.E., 12-bit integrating A/D; programmable gains of 1, 10, & 100 \$159/259
 ADA100 Single-channel, differential input, 12-bit integrating A/D; S-bit D/A output; programmable gains of 1, 10, & 100. Plus 10 digital I/O lines. \$219
 ADA300 S-channel 8-bit 25 μ s A/D; single 8-bit D/A; 24 TTL digital I/O lines \$199
 DA600/DA700 Fast-settling 2/4/6/8 -channel 12-bit D/A; double buffered \$192/359
 DG24/48/72/96-line TTL/CMOS-compatible digital I/O cards; 8255-based; Opt. buffers and pull-up resistors \$110/256
 TC24 Five 5-MHz timer/counters; uses powerful AM95 13A chip; 24 digital I/O lines from 'ML/CMOS-compatible PPI chip \$218
 ATLANTIS/PEGASUS High-performance data acquisition/analysis software; foreground/background operation; maximum 25-KHz rate; supports hard disk streaming; pull-down windows; interactive graphical data analysis; hypertext help \$150/250

Custom/OEM designs on request!

Real Time Devices, Inc.

rtdd 820 N. University
P.O. Box 906
State College, PA 16804

Phone: 814/234-8087

FAX: 814/234-5218

Reader Service #182

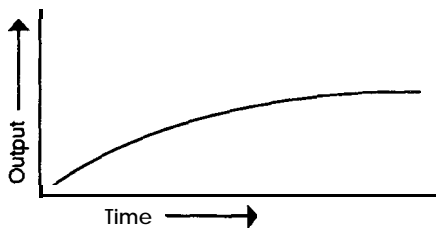


Figure 4e— Rarely is a heavily overdamped response required. This system would be very sluggish.

tional gain can be set. Temporarily disable the integral and derivative terms, and set the gain to a low value, about one or two. Start up the control algorithm, and watch for the process to settle. Remember that it probably won't settle precisely at the set point; just wait for it stop moving. If the process refuses to reach equilibrium with this gain, you may have to back off on the controller's bandwidth some more.

If the process does settle, observe the system's response to a set point change of about 10%. If no oscillation is noted, increase the gain in a 1-2-5 sequence and bump the setpoint gain until a persistent oscillation is noted. Then, drop back two steps in gain. This value should give a good compromise between best performance and stability in the face of changing conditions.

Once the proportional gain is set, the integral term is next. Note that if the proportional term is high, and a little error is tolerable, integral control may not be needed. If this is the case, try to test the system under the widest variation of circumstances you can think of to make sure that the performance is acceptable.

If integral control is used, again start with a low gain value and turn it up in the 1-2-5 sequence outlined above. When the set point is shifted, more overshoot will be noted than when the system had only proportional gain. The overshoot should quickly die out—three cycles is a good rule of thumb—and the steady-state error should rapidly converge to zero. Sometimes it is impossible to achieve rapid convergence without overshoot. If required, the proportional gain can

be backed off to allow more integral action to be used. Proper automated control action is a compromise between oscillation, overshoot, and convergence. Taking careful notes while varying the gain constants should allow you to quickly select a good set of values.

Once a balance between proportional and integral action is reached, the use of derivative action can be evaluated. Unlike the other two terms, the more derivative action the better until noise becomes a problem. The derivative control term can simply be turned up in the normal sequence until "nervousness" of the end element is observed. Turning the gain constant down two steps should then result in good response.

One thing to watch out for in systems with significant derivative action is for factors which might cause the noise to go up. The classic example is a servo system which uses a potentiometer as a feedback element to sense the position of a rotational element. After a period of use, mechanical wear can cause the pot to become noisy, which will drive a previously stable system bananas! Because of this, non-wearing optical or magnetic sensors are often used, or the mechanical sensor can be replaced at regular intervals.

THE RESULTS

Now that you've conquered the challenge of control theory, you'll have more time to lounge in your hot tub. Even better, you can now rest assured that no matter how the weather changes, or how many friends come over, you'll always have 110°F water. This should make all your hard work worth it. ❖

Tom Mosteller is a field application engineer for a major linear IC manufacturer.

IRS

262 Very Useful
263 Moderately Useful
264 Not Useful

Using C For Embedded Control

Building a 6805-Based Darkroom Timer

FEATURE ARTICLE

Ashok Patel
& Walter Banks

It's tough to get a consistent black-and-white print of the Orion Nebula. This was a couple years ago when Mike asked Ashok if he had any ideas. Ashok always has solutions and the Darkroom Timer project was born.

This project was designed to solve a simple enlarger exposure timer problem and wound up being used to test a C compiler for use in embedded computer applications.

The Darkroom Timer is designed to be a general-purpose electronic timer with both a switched output and alert buzzer. The basic resolution of the timer is 0.01 seconds. The design has such advanced features as last entry recall, manual toggling of the switched output, and use as an electronic count-down timer with audible warning.

This project is typical of many embedded computer products. It is a low cost, simple to use consumer product designed to fill a very real need. The timer is a technological update from the days when a 555-based analog timer might have been used to control a relay-driven light. It uses a low-cost 8-bit microcomputer to scan both the keyboard and the seven-segment LED display. The software incorporated into the darkroom timer is also typical of real-time consumer products. The software must respond to a timer interrupt which arrives at a rate of 5,000 interrupts per second, multiplex an LED display, and scan a keypad. Finally, like most consumer products, the processor in the Darkroom Timer has limited RAM, ROM, and CPU resources which must be taken into account during development.

Photo 1 shows the final prototype of the Darkroom Timer in its case. The

case size is about 5 by 6 inches. The electronics in the timer were assembled on a small (3" by 6") piece of perforated board. Photo 2 shows the top of the electronics board.

DESIGN CONSIDERATIONS

The choice of processor was originally fairly simple. The Motorola 6805 is widely used in process control, data

logging, consumer, industrial and automotive applications, and is cheaply available at most electronics stores.

To keep the hardware to a bare minimum, the keyboard and display were to be scanned with software. It was later while the software was being written that we decided the LEDs could be scanned as part of the timer interrupt routine.



Photo 1—The prototype darkroom timer uses a 6-digit LED display and a 13-key keypad.

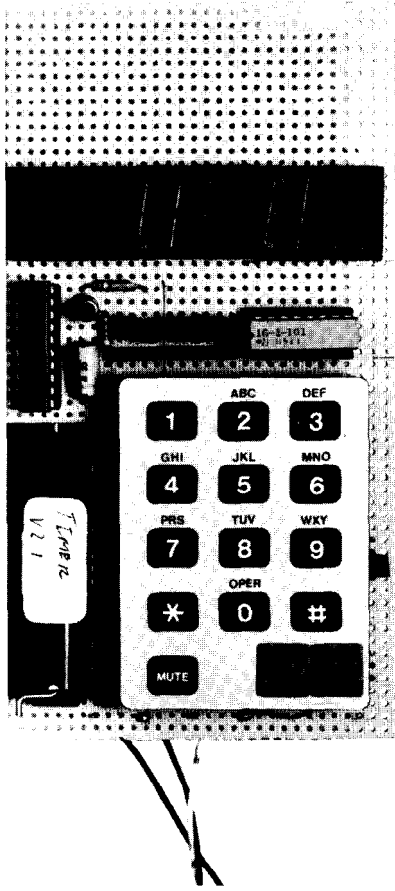


Photo 2—The MC68705 is installed in a ZIF socket to make insertion and removal easy.

The speaker is a half-inch speaker of about 200Ω used in many applications. It is also driven by a software tone generator through a single parallel line.

The interface to control the electronic relay circuit uses a triac and an optically coupled triac driver to isolate the electronics in the Darkroom Timer from the 115-volt line voltage. This circuit is essentially the same as that used in many solid-state relays, such as the common industrial Opto-22 series.

As we were developing and testing a C compiler for the 6805 family of embedded processors, we pressed the Darkroom Timer into service as a test bed for comparing a project implemented in hand-written assembler with code produced by a C compiler.

OPERATION SPECIFICATIONS

The heart of the Darkroom Timer is a single-chip microcomputer, the

MC68705U3. This microcomputer has 3776 bytes of EPROM, 112 bytes of RAM, 24 bidirectional and eight input-only parallel lines, a timer/counter, and an on-chip clock generator. A single +5-volt supply powers the system. Drivers, clock crystal, and bypass capacitors are the only external components used by the Darkroom Timer. The display is driven by ports A and B, the keypad is scanned by ports C and D, and two of the outputs of port C are used to control the light and buzzer circuits.

The crystal frequency is 4.0 MHz. This makes it very easy to use the built-in timer to interrupt the program 5,000 times per second. The interrupt handler controls the timing, display multiplexing, and the speaker. The keypad is scanned by a software foreground loop.

THE DISPLAY

The display is a 1", 6-digit, 7-segment common-anode display. The red display was chosen so that it will not expose photographic paper. The microcomputer is responsible for scanning the display, and for decoding the digits to be displayed into 7-segment outputs. The amount of current required for the large display is greater than the amount the processor can deliver on its parallel lines, therefore high-current buffers are the only external components needed to drive the display. The segments are driven by a 74LS244 buffer through port B of the 6805. The common anodes are driven by transistors in a voltage follower configuration and are controlled by port A of the 6805. The transistors are necessary because when all seven segments are lit, the display draws about 140 mA. A resistor (R22) is used to current limit the display.

THE KEYPAD

The keypad is a telephone-style keypad. It has 12 keys (0-9, #, and *) in a 3 x 4 matrix and a separate mute key. The processor scans the 3 x 4 keypad with the columns connected to port C and the rows connected to port D. The mute key is connected to bit 4 of port

D and is pulled up to +5V through a 10k resistor. When this key is pressed, the bit goes from 1 to 0.

DRIVING THE REAL WORLD

The light and speaker are controlled by two bits of port C. The speaker is driven by a transistor switch driven from an output bit on the processor. The schematic shows a 1k potentiometer used to control audible volume. Rather than a potentiometer, though, our prototype uses a 470Ω fixed resistor to produce a reasonable sound level.

The light control line is used to drive an LED in an optically isolated triac driver chip. The chip then drives a triac to operate the 115-volt AC line. This combination of the optically isolated triac and triac switch is really a discreet version of widely used solid-state relays.

THE SOFTWARE

The program is a classic real-time control system. The software is divided into two parts: The foreground task scans the keypad and interfaces the human user to the timer. The interrupt handler counts down the elapsed time, scans the LED display, and generates the tone for the speaker.

Communication between the foreground and interrupt handler is accomplished through global variables. A semaphore is used to ensure both processes do not attempt to modify the global variables at the same time.

The interrupt handler is invoked by the interval timer 5,000 times per second. This rate is further divided by ten in software to provide a 500-digit-per-second scan rate of the LED display. It is divided once more by five in software to provide 1/100th of a second resolution of the elapsed time.

The elapsed time is stored in an array of six BCD digits for ease of LED display and count down purposes. Routines in the interrupt handler decrement the 6-digit BCD number.

The LED display routine is executed 500 times per second. Each time it is executed, a single digit is displayed. The array of BCD digits is

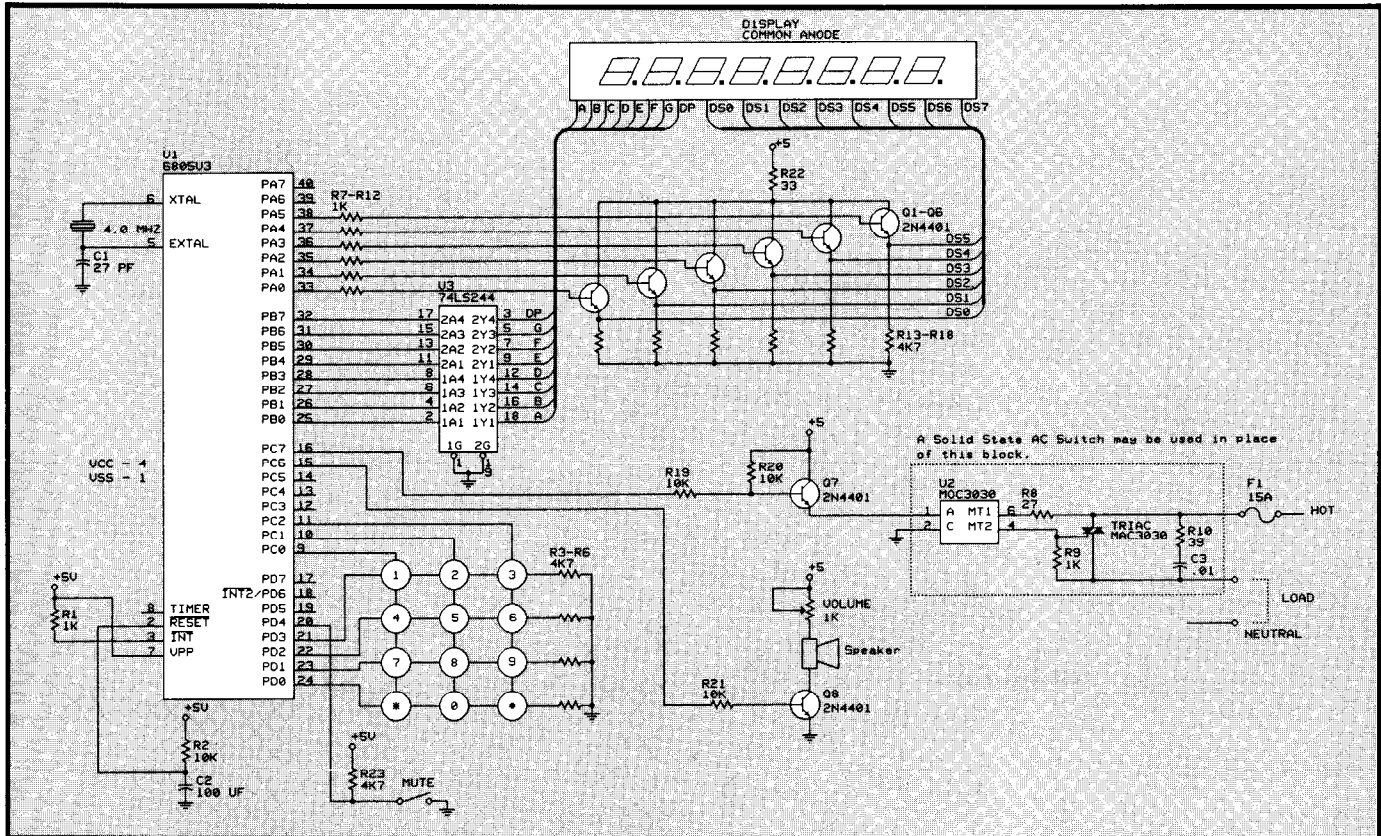


Figure 1 - With its internal RAM and EPROM, the MC68705 simplifies circuit design, requiring just a keypad, display, speaker, and solid-state relay to make up the complete darkroom timer.

R-535 Prototyping Board Plus R-WARE

A Complete System for Developing Embedded Control Applications

Board includes: power supply, 80535 processor (enhanced 8052 with 3 timers plus watchdog timer and 12 interrupt sources at 4 programmable priority levels), up to 256 k on-board memory, Eprom burner, RS-232 serial port at 9600 Baud, 28 digital I/O lines, 8 analog input lines, 2-1/8" by 6-1/2" breadboard with 8 pushbuttons, 8 toggle switches, 16 LEDs, 2 numeric displays.

R-Ware includes: ROM resident monitor program and PC-based integrated menu-driven software for edit, assembly, PC-to-board communication, download and debug.

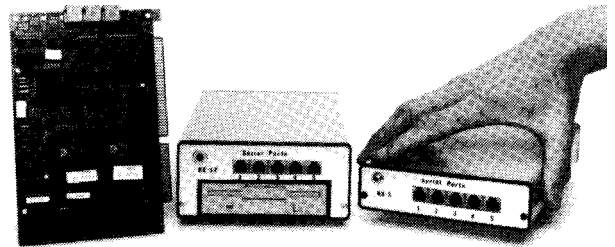
Plus comprehensive user's manual and control experiments with example software.

Prices start at \$395.-

RIGEL CORPORATION

P.O. Box 90040
Gainesville, FL 32607
(904) 373-4629

Get on the PC BUS



ROM or Disk based AT Systems Cards \$299, Systems \$399

It's easy to run your compatible applications on our single board computer! Develop code on a PC, and follow our ten easy steps to place your .exe files and DOS in ROM.

PU Card: V50 CPU, 8086 Code Compatible
1 MB Ram, 256kB Rom, 4.5" x 7"
5 Serial Ports... CMOS (2 watt)

Expansion: Backplanes for PC/AT cards
Piggyback card with: Floppy,
SCSI, Printer, and Keyboard ports

software: BIOS. Utilities. Monitor, & Source code



303-444-7737 fax 303-786-9983
655 Hawthorn Ave., Boulder CO 80304

USING C IN EMBEDDED APPLICATIONS

The C language was originally created at Bell Labs as an alternative to using assembly language as the development vehicle. High-level languages have existed for embedded systems for a number of years. Advocates point out the ease of use and reliability of systems implemented by a high-level language. Some form of BASIC or Tiny C is available for many of the microcontrollers in common use. Both have been plagued by code inefficiency and high execution times. Even so, many applications have benefited from the use of a high-level language.

As single-chip microprocessors become more ASIC-like with an embedded processor core and peripheral parts surrounding it, software tools need to perform consistency checks with the predefined target environment. The C6805 compiler checks the system hardware definitions against the C source. It will not attempt to violate the system constraints; a write to ROM, for example. Differences between various members of the 6805 family are specified through compiler-specific directives called #pragmas.

```

0040 0041 0042      char i,j,k;
                    main()
                    {
0100 B6 41 LDA $41      i= j + k;
0102 BB 42 ADD $42
0104 B7 40 STA $40
                    }

```

Listing I—C program in C6805 to add two numbers and store the result.

Within the last few years, cross-development platforms used in creating application obj code have improved substantially. This improvement has given the software tool suppliers the ability to translate high-level languages into machine code that is nearly as efficient as the best hand-written assembler.

In simple statements, C is a compact way of describing a problem. Take a look at the example in Listing I. The variables i, j, and k declared as chars (unsigned 8-bit quantities) are placed by the compiler at locations 40 through 42. The code

itself compiles starting at ROM location 100. The generated code looks familiar and so it should.

The example in Listing I demonstrates the primary reason that most people use high-level languages. The compiler translates easily understood statements into tight machine language representations of the code. Listing II shows a code

```

0000                                #define c0 0
0001                                #define c1 1
0005                                #define c5 5

0040 0041      char i,j;
                    main()
                    {
0100 B6 40 LDA $40      j = i + c5;
0102 AB 05 ADD #$05
0104 B7 41 STA $41

0106 B6 40 LDA $40      j= i t c0;
0108 B7 41 STA $41

010A 4C      INCA      j = i + c1;
010B B7 41 STA $41
                    }

```

Listing II—C optimization in syntactically identical statements.

fragment where three constants are defined at the beginning of the program and each constant is used in syntactically identical statements. In the first statement j is assigned the value of i plus the constant c5 which is defined as having a value of 5. As expected, the machine code is a "load, add, store" sequence. The next line adds 0 to the current value of i and stores it in j. The C6805 optimizer recognizes the 0 and simply generates a load/store code sequence. The final line adds 1 to the current value of i. The optimizer does two things. First, it sees that the current value for i is already in the accumulator and, it recognizes an increment as being more effective than adding 1.

The kind of code generated in Listing II is typical of the code generated by compilers using current optimization technology: tight, effective, and fast code that is consistent with well-written assembler code.

indexed by a variable in this routine. The current BCD digit is used to access a table to decode the digit into its seven segment representation. This value is output to the display segment port B. The index then selects the digit to be displayed by turning on the appropriate bit in port A.

An audible tone is generated when needed by toggling the speaker output line every time through the interrupt handler thereby generating a 2,500-Hz tone. The tone length is established by setting a global variable to the number of half cycles of tone to be generated. If the tone length is zero, the speaker is left off.

The foreground task scans the keyboard and sets the BCD value to the time selected by the user and initiates or stops the count down routines. The foreground task also has a command decoder that processes the nonnumeric keys, mute, #, and *.

The Darkroom Timer was originally programmed in assembler and later converted to C using the Byte Craft C6805 compiler to validate the competitiveness of assembly language code versus the compiled code. The assembly language algorithms were implemented in C, and the resulting object code compared. The C6805 did indeed generate competitive code, and

timing-sensitive interrupt service routines were effectively written in a high-level language. For the curious, the initially compiled code was six bytes longer than the assembly language version. [Editor's Note: Software for this article is available on the Circuit Cellar BBS and on Software On Disk #17. See page 108 for downloading and ordering information.]

USING THE DARKROOM TIMER

The time for the enlarger to turn on is set by using the number keys 0-9 on the numeric keypad. The numbers are recorded from right to left. As

digits are entered, the previous digits shift left by one.

After setting the time, the # key is used to start the timer. When the # key is first pressed, the enlarger turns on, and the timer starts to count down. If the # key is subsequently pressed, the timer stops, the enlarger turns off, and the timer displays the time remaining. If the # key is pressed again, the timer restarts at the time remaining and continues to count down. When the timer reaches zero, it turns off the enlarger AC power and sounds a tone (if the speaker is enabled). A new time may now be entered, or if the # key is pressed without a new time entered, it restarts the timer using the previous setting, thus, simplifying the process of printing multiple copies of a print that have the same exposure time.

The * key is the preview key. Pressing it once turns on the enlarger, and pressing it again turns the enlarger off. This is not a timed exposure but is used to focus the enlarger.

The mute key is used to enable or disable the speaker. The timer starts

up with the speaker disabled. That is, when the timer expires, it does not sound a tone. If the tone is desired, press the mute key. The speaker sounds to indicate that the end-of-time tone has been enabled. This is useful when developing films that need a count-down timer with an audible warning when exposing the film to the various developing chemicals.

SUMMING UP

The Darkroom Timer design is viable even as a commercial product. The original prototype was implemented using a MC68705U3 microprocessor. This part is readily available on the surplus market for a few dollars. If this design were to be developed into a commercial product it would be advisable to use today's latest 68HC05 processors. The C compiler used to implement the darkroom timer is designed to generate ROMable code for all versions of the 6805 family.

The Darkroom Timer could be extended to perform other darkroom functions such as controlling the temperature of the various developer solutions.

What have we learned? Certainly high-level languages are now becoming practical in producing code competitive with well-written assembly code for single-chip controllers.+

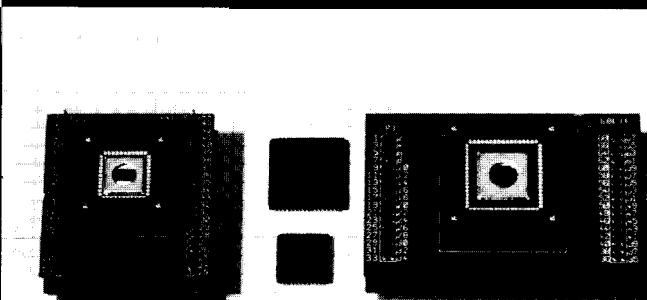
Ashok Patel is the president of Softart Microsystems Inc. where he does hardware and software designs using microcontrollers for the communications industry. He holds a BMath degree from the University of Waterloo (specializing in Computer Science).

Walter Banks is the president of ByteCraft Limited, a company specializing in software development tools for microcontrollers.

IRS

- 265 Very Useful
- 266 Moderately Useful
- 267 Not Useful

PLCC EPROM ADAPTERS



Program PLCC EPROMs on your DIP programmer. The PLCC sockets are live-bug auto-eject style. They provide positive alignment of the device with a push-in/pop-out mechanism.

2784 thru 27512.....	PA28-32	\$90.00
28 DIP to 32 PLCC		
27010 thru 27040	PA32-32	\$90.00
32 DIP to 32 PLCC		
27210 thru 27280	PA280-44	\$115.00
40 DIP to 44 PLCC		
87C552 PLCC	PA552-68	\$185.00

CALL or FAX for full device list!

CALL (315) 478-0722 FAX (315) 475-8460

Logical Systems Corporation
P.O. Box 8184, Syracuse NY 13217 USA

Reader Service #155

ProControl!

Modular data acquisition and control for your IBM PC

At last, an industrial quality system that can start small and easily expand! Just insert the Host Adapter into your PC and then add only the I/O modules you need.

IBM PC Host Adapter \$129.95

Interfaces IBM PC's to ProControl bus I/O modules. Includes 24 digital I/O lines built-in and can connect directly to Opto-22 PB-24 I/O racks.

AIN-8 16 A/D Module \$129.95

Sixteen 8-bit analog-to-digital conversion channels, 50us conversion time (20us optional), on-board voltage reference and trim pots.

AOUT-84 D/A Module \$99.95

Four (expandable to six) 8-bit digital-to-analog conversion channels.

Industrial Card Cage \$199.95

Safely house 8 I/O modules (expandable to 24 slots).

More I/O modules are available. Call for our FREE catalogue today!

(404) 352-47 88

ADS Advanced Design Solutions

1920 Moores Mill Road
Atlanta, GA 30318

Reader Service #102

The Furnace Firmware Project, Part 3

Tight Code Meets The C Monster

FIRMWARE FURNACE

Ed Nisley

Now that the Furnace Firmware project has a display and keypad, we need a way to measure temperatures and record the results. The topics this time around are analog input, timekeeping, and non-volatile RAM.

The RTCIO board has most of the hardware needed for this column, so I borrowed the schematics from that manual. I also wired up a few new parts to add nonvolatile memory and some error checking. So far, an RTC-PROTO board hasn't wound up in the stack: somehow, Jeff designs in room for just one more part even on the most crowded board!

THE TIME AT THE CHIME

In the last column I described how the 8031's Timer 0 interrupt handler measures time in 5-ms ticks, paces the keypad scanning, and sets the beeper tone. However, creating a calendar from a 5-ms tick is rather difficult. And, of course, if (when!) the power blinks off, you start all over from year zero...

A better choice is a clock/calendar chip with its own battery and crystal oscillator. Once you set the correct date and time, the hardware will keep it current forever thereafter. The RTCIO board uses the Oki M6242B, which, in addition to its other tricks, handles (most) leap years correctly.

does not count the year's two high-order digits; be sure to take heed if you're developing truly bullet-proof code.

Figure 1 shows both the clock/calendar chip and the RTCIO address circuitry used by the rest of the board. The default RTCIO board base address is E000, with the four chip selects spaced at 0010 hex intervals. Remember that the RTC-LCD board is decoded at E080, so there is no conflict.

A 3-volt lithium cell supplies backup power through a blocking diode. According to the M6242 specs, that 165-mAH cell can power it continuously for a year or two; I really don't expect a power outage of that duration!

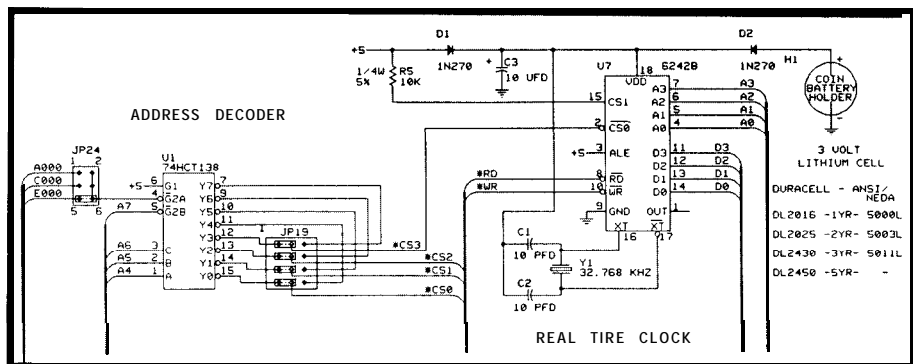


Figure 1 — The clock-calendar chip and the RTCIO address circuitry used by the RTCIO board. The default RTCIO board base address is E000h, with the four chip selects spaced at intervals of 0010h.

The M6242 thinks the year 2000 is a leap year, which is true. A year is a leap year if it is evenly divisible by four, except if it's divisible by 100, unless it's divisible by 400. Got that? The year 1900 wasn't a leap year, 2000 will be, but 2100 won't. The M6242 will fumble the year 2100 because it

Another RTCIO option is a DC-to-DC converter to power the ADC and DAC chips from a single +5-VDC supply. That converter, a MAX633, also provides a "power failure" warning signal to the M6242 which prevents accidentally writing to the registers during a power failure. I'm not

a bit easier than calibrating a thermistor?

Best of all, the cable between an LM-335 and the rest of the system can be a simple two-wire affair! Modular phone cable is cheap and readily available; a run of 4-conductor cable puts a pair of sensors halfway across the house.

There are other chips in the same family, such as the LM34 and LM35 Precision Sensors, which report directly in degrees Fahrenheit or Centigrade (Celsius) times 10 mV, but these chips require an additional negative supply voltage to produce the correct output for temperatures below zero degrees. Although I don't intend to freeze my furnace, I do plan to include an outdoor air temperature sensor and winternights can be pretty cold around here. Given that carbon dioxide freezes at 194 K and tin melts at 505 K, the LM335 and ADC0808 can handle the expected range of temperatures, though.

Photo 1 shows an LM335 mounted in a nylon pipe clamp. The recess machined into the clamp holds the flat side of the TO-92 package tangent to the pipe and a blob of thermal grease ensures good contact with the copper pipe. I strapped a modular phone connector onto the clamp, connected the proper two wires to the LM335, and insulated the leads with a dab of silicone rubber.

The RTCIO board can hold either of two pin-compatible analog-to-digital converters: the 8-bit ADC0808 or

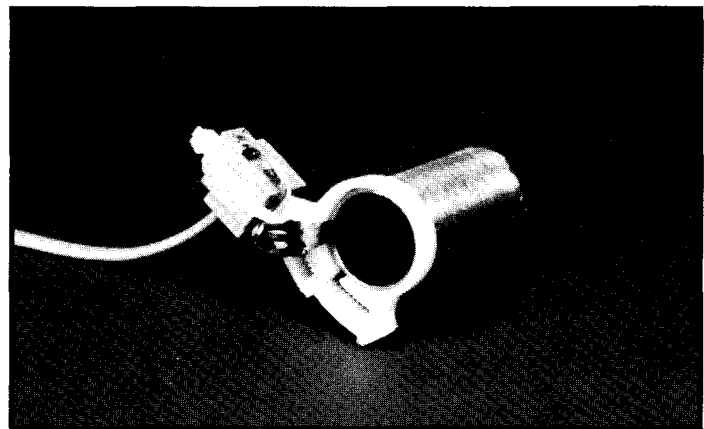


Photo 1 - An LM335 mounted in a nylon pipe clamp. The recess machined into the clamp holds the flat side of the TO-92 package tangent to the pipe, so a blob of thermal grease ensures good contact with the copper pipe. A modular phone connector carries two wires to the LM335.

```

;-----
; Fetch current clock-calendar registers
; These occupy a static array for simplicity, returned in DPTR
; The clock runs while we read the regs, so check for changes

TmrGetTOD   PROC
PUBLIC      TmrGetTOD

    PUSH    0          ; save a bystander

    GetCWord ClockBase ; fetch clock chip port
    MOV     P2,DPH
    MOV     R0,DPL
    MOV     DPTR,#ClockRegs ; point to RAM copy
    MOV     B,#16       ; num of regs to capture

L?fetch     MOVX    A,@R0      ; fetch a reg
            MOVX    @DPTR,A   ; store it
            INC     DPTR      ; tick pointers
            INC     R0
            DJNZ   B,L?fetch

            MOV     A,R0      ; back up to seconds port
            ADD    A,#-16     ; (can't borrow!)
            MOV     R0,A

            MOV     DPTR,#ClockRegs ; check seconds for change
            MOVX   A,@R0      ; new value
            MOV     B,A
            MOVX   A,@DPTR    ; old value
            POP     0         ; restore before loop
            CJNE   A,B,TmrGetTOD ; try again...

            RET

TmrGetTOD   ENDPROC

```

listing 1 - This code extracts the OKI M6242 clock/calendar registers into External RAM. Because the clock continues to run while being read, the registers could change. If the 'seconds' register has changed, the code rereads all of the registers to ensure a valid set.

using the MAX633, so if the lights go out when you're setting the time, remember to check the results.

Listing 1 shows how to read the current time from the M6242. *[Editor's Note: Software for this article is available on the Circuit Cellar BBS and Software On Disk #17. See page 108 for downloading and ordering information.]* One of the registers includes a "hold" bit that prevents the registers from changing during a read, but I don't want to write to the chip unless necessary. So the code reads all 16 registers, then reads the "seconds" register again. If both values agree, the other registers are valid; if not, the code tries again.

The demo program CDEMOTOD . HEX allows you to set the date and time, then displays the current status once a second. Because the clock chip continues to run even when the RTC boards are turned off, you should find that the time is (more or less) correct

even after a number of days without power.

TAKING A TEMPERATURE

There are many ways to convert temperature into voltage, but one of the better is the LM335 Precision Temperature Sensor. In a nutshell, the chip's output voltage is equal to the temperature in Kelvin times 0.01 volts (10 mV/K). A room temperature of 70°F is 21°C or 294 K, so the LM335 output voltage is 2.94 volts. Now, isn't that

the 10-bit SDA0810. Both chips are pin-compatible, I can start with the ADC0808 and substitute the SDA0810 when I get to the final testing stage.

Remember that one good experiment is worth 1000 expert opinions. As Steve found out with his data logger (CIRCUIT CELLAR INK #15), some-

Because the chips are pin-compatible, I can start with the ADC0808 and substitute the SDA0810 when I get to the final testing stage.

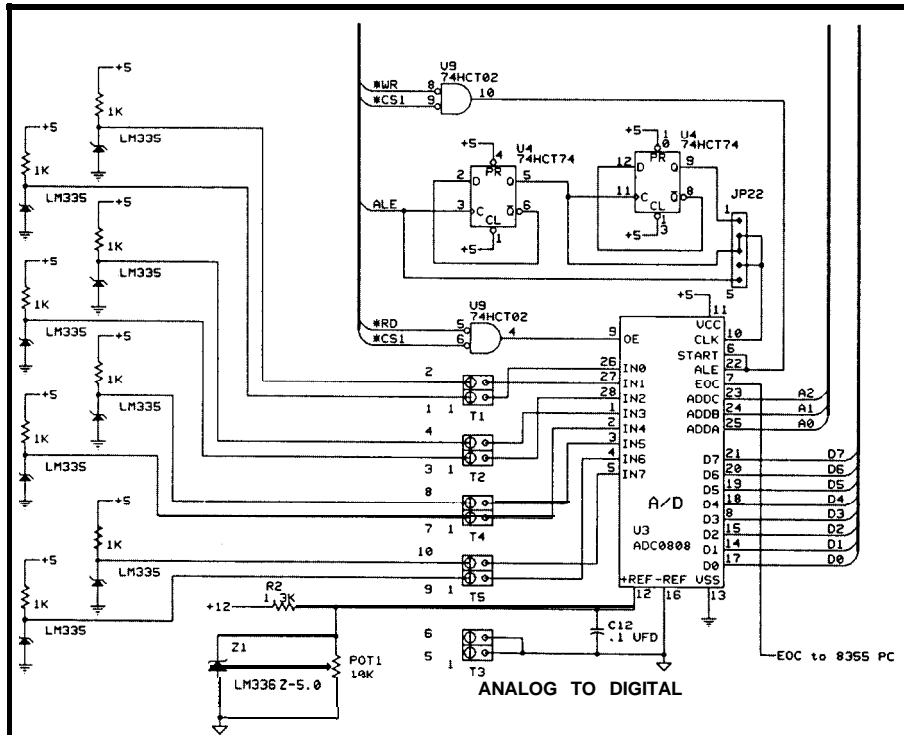


Figure 2—The ADC0808 analog-to-digital converter includes an eight-input multiplexer and an eight-bit successive-approximation converter.

urement error. Figure 2 shows the circuitry required for the ADC0808 on the RTCIO board.

The input voltage range is 0.0 to 5.0 volts, so the 256 binary values produced by an 8-bit converter such as the ADC0808 are 19.5 mV apart. The LM335 scale factor of 10 mV/K means that the smallest resolvable temperature change is nearly 2K, which translates into 3.6°F. The difference of a heating zone's inlet and outlet temperatures can be in error by about 7.2°F, which is most likely a significant fraction of the true temperature drop.

The SDA0810 converter provides two additional bits that give a voltage resolution of 4.8 mV and a temperature resolution of 0.48 K (0.8°F). That reduces the zone measurement error to about 1.6 degrees, which is probably adequate for this application.

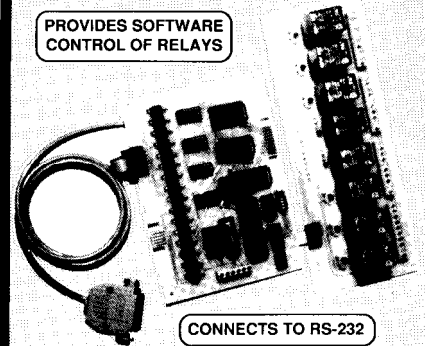
times you just have to measure reality and then decide what to do!

As far as conversion time goes, the ADC0808/0810 chips use successive approximation, so the total time doesn't depend very much on the input voltage. The ADC0808 specs imply a maximum of 75 clock cycles per conversion, which, at the nominal 922-kHz clock on the RTCIO board, gives a typical delay of about 80 microseconds. The End-of-Conversion pin goes low when a conversion starts and returns high when data is valid.

Under normal circumstances, the EOC signal connects (through an inverter) to either of the two 8031 External Interrupt pins. However, because the RTCIO board has an 8255, I wired the EOC signal to one of those input bits to read it directly. This keeps the INT pins available for other, more critical, uses.

RELAY INTERFACE

PROVIDES SOFTWARE CONTROL OF RELAYS

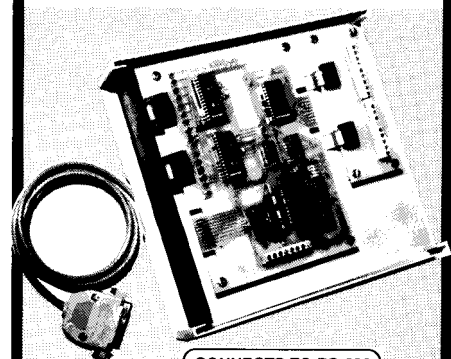


CONNECTS TO RS-232

AM-16 RELAY INTERFACE \$89.95

Two 8 channel relay output ports are provided for control of up to 16 relays (expandable to 126 relays using EX-16 expansion cards). Each relay output port connects to a relay card or terminal block. A variety of relays and relays are stocked, call for more info. FID-9 REED RELAY CARD (8 relays) \$49.95 FID-8 RELAY CARD (10 amp SPDT 277 VAC) \$69.95 EIX-16 EXPANSION CARD (16 channel) \$59.95

ANALOG TO DIGITAL



CONNECTS TO RS-232

ADC-16 (16 channel) \$99.95

Input temperature, voltage, amperage, pressure, energy usage, energy demand, light levels, joystick movement and a wide variety of other types of analog signals. Inputs may be expanded to 126 status inputs using the ST-32 expansion cards and up to 112 relays may be controlled using EX-16 expansion cards. A analog inputs may be configured for temperature input using the TE-6 temperature input card. (enclosure, terminal block and cable sold separately) ST-32 STATUS EXPANSION CARD \$79.95 Input on/off status of switches, thermostats, relays, security devices, smoke detectors, HVAC equipment and hundreds of other devices. The ST-32 provides 32 status inputs (opto isolators sold separately). TE-8 TEMPERATURE INPUT CARD \$49.95 Includes 8 solid state temperature sensors and 8 calibration trimmers. Temperature range is minus 76 to 145 degrees F. Very accurate.

FULL TECHNICAL SUPPORT. Provided over the telephone by our staff. A detailed technical reference manual is provided with each order including programming examples on disk in Basic, C and Assembly Language.

HIGH RELIABILITY. engineered for continuous 24 hour industrial applications.

Use with IBM and compatibles, Tandy, Apple and most other computers with RS-232 or RS-422 ports. All standard baud rates and protocols may be used (50 to 19,200 baud) default is 9,600 baud 8 data bits, 2 stop bits, no parity

Use our 600 number to order free information packet. Technical Information (614) 464.4470.

24 HOUR ORDER LINE (600) 642-7714
Visa-Mastercard-American Express-COD

ELECTRONIC ENERGY CONTROL, INC.
360 South Fifth Street, Suite 604
Columbus, Ohio 43215

Reader Service #131

However, monitoring this signal through the 8255 introduces an interesting complexity. The 8031 ALE signal normally pulses once per machine cycle, so it runs at 1/12 the oscillator frequency. However, any External Data Memory access suppresses the ALE signal for one cycle. Therefore, you should not code a tight wait loop with an external memory access because that will provide a very irregular ALE signal and confuse the ADC0808.

Listing 2 shows the assembler code required to convert an input and read the result. As is usually the case, most of the code is devoted to handling the unlikely case of a missing End-of-Convert signal! Loop setup requires about 65 microseconds, so there should be only two or three iterations of the

16-microsecond loop before EOC goes active. ADCMaxDelay sets the maximum number of iterations the code will wait before it bails out and returns an error.

The alert reader will notice that Listing 2 is suspiciously different from the listings so far. The routine name starts with an underscore, unlike the other routines it uses registers with gleeful abandon, and it even returns a value in R2 and R3. What's going on here?

At the beginning of this project, I promised I was going to experiment with C for 8031 processors. I'll say more about the results later on, but for now, the code in Listing 2 is designed to be called from an Avocet C program. If you're using assembler, just make sure that you save the ap-

propriate registers and return the value correctly.

In this case, the return value is a two-byte integer with the ADC's output in the lower byte. If I substitute an SDA0810, the ten bits will occupy both bytes; the calling routine must know that the value can now go up to 1023 instead of just 255. A value of FFFF flags a timeout error for either kind of ADC.

The demo program CDEMOADC.HEX reads all eight channels and displays the results as a 16-bit hex word. It assumes you have an ADC0808, so it won't read any additional bits from the converter. You will need to make a few straightforward changes to the conversion routine to extract the two additional bits from an SDA0810; I heroically resisted the temptation to use another configuration variable!

THINGS REMEMBERING

Most single-board computer systems start afresh every time they are reset. The initialization routine clears RAM, presets the outputs, and runs the program "from the top" with no memory of what occurred before the reset. Generally, this works out OK, but there are some situations where you really need to remember events more or less forever.

For example, the LM335 temperature sensors each have slightly different calibration constants. It Would Be Nice If you didn't have to enter the values after every reset!

One way to achieve permanent memory is to bum the constants into EPROM along with the program. The Furnace Firmware stores I/O addresses, bit locations, and so forth in just this way, because these values generally don't change. However, a calibration "constant" may need adjustment due to component replacement or drift. Rebuming the EPROM each time a change is necessary could pose a problem, particularly because you have to shut the system down to replace the old EPROM with the new one.

A better solution uses nonvolatile RAM to hold "variable constants" and

```

;-----
; Return analog voltage for aiven channel
; Presumes EOC is available on input port pin
; WORD ADCGetChannel(int Channel);

_ADCGetChannel PROC
    PUBLIC      __ADCGetChannel

    GetCWord   ADCBase      ; figure starting channel
    AddDPTR   R5
    CLR       A              ; any value will do
    MOVX     @DPTR,A        ; start conversion

    GetCWord   ADCMaxDelay ; EOC timeout
    MOV       RO,DPH
    MOV       R1,DPL
    GetCData   ADCEOCMask  ; EOC bit location
    MOV       R2,A
    GetCData   ADCEOCFlip  ; EOC bit inversion
    MOV       R3,A
    GetCWord   ADCEOCPort ; EOC port address

L?wait      MOVX     A,@DPTR    ; fetch EOC bit
            ANL     A,R2      ; isolate it
            XRL     A,R3      ; flip if needed
            JNZ     L?ready   ; if not zero, finished!

            DecRR   R0,R1     ; tick timeout count
            MOV     A,R0
            ORL     A,R1
            JNZ     L?wait    ; if not zero, continue

            MOV     R2,$FF    ; force error flag
            MOV     R3,$FF
            SJMP   L?done

L?ready     GetCWord   ADCBase ; fetch result
            MOVX     A,@DPTR
            MOV     R2,#0     ; high byte is always zero
            MOV     R3,A      ; low byte is the reading

L?done     RET

__ADCGetChannel ENDPROC

```

listing 2—The ADC0808 analog-to-digital converter provides an End-of-Conversion signal that is wired to an 8255 input bit. This code is called as a subroutine from a C-language program. The multiplexer channel is passed in general register R5.

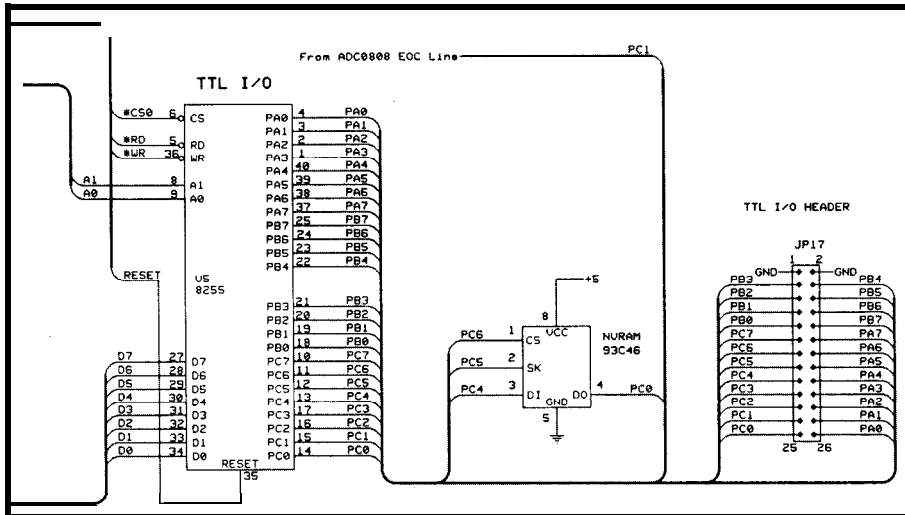


Figure 3—The RTCIO's 8255 makes a convenient interface to the NMC9346 EEPROM chip.

other important data. An NVRAM (or EEPROM) maintains its values without an external power supply, unlike battery-backed RAM, so you don't need to include additional circuitry. NVRAMs come in all shapes and sizes, but, for this application, I picked the NMC9346 1024-bit serial EEPROM because it is physically small (an 8-pin DIP), electrically simple (+5 VDC power), and, best of all, I've used it in another project so I know just how it works.

Figure 3 shows an NMC9346 EEPROM chip connected to the RTCIO's 8255. Photo 2 shows how I mounted the chip and socket upside down in the spot normally used by the inductors for the DC-to-DC converter on the RTCIO board. Hot-melt glue to the rescue!

Internally, the NMC9346 has 64 16-bit data registers. Externally, there are only four signal pins: Chip Enable, Serial Clock, Data In, and Data Out. The keyword "serial" should warn

```

;-----
; Send command and address from A
; Presumes P2:R0 points to 8255 port
; Leaves chip enabled, ready for data I/O
; ... so you have to finish the command sequence
; Drops NVR DI line after lowering the strobe

NvrSendCmd PROC
PUBLIC NvrSendCmd

    PUSH ACC ; save command byte
    SetPBit CS ; enable the chip

    CLR C ; pre-start bit
    CALL NvrBitOut
    SETB c ; start bit
    CALL NvrBitOut

    POP ACC ; recover the byte

NvrSendData EQU $
MOV B, #8 ; eight cmd t address bits

L?next RLC A ; get next bit into C
CALL NvrBitOut
DJNZ B,L?next

ClrPBit DI ; drop data to RAM RET

NvrSendCmd ENDP

```

Listing 3—The NMC9346 EEPROM uses a one-bit serial interface. This code transfers a command and address from the 803 J accumulator to the NVRAM. The NvrSendData entry point transfers a data byte to the chip; it is called twice to write a 16-bit entry.

\$109

(without memory)

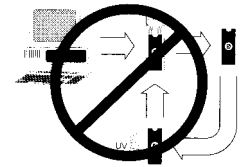
\$129

(32K x 8 SRAM)

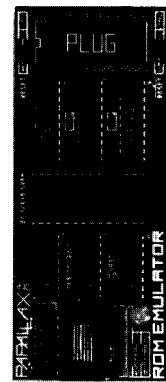
\$149

(32K x 8 NV SRAM)

Avoid the hassles of programming EPROMs



A ROM Emulator can greatly reduce the time spent writing and debugging ROM code



Emulates 2764, 27128, and 27256

Plugs into target ROM socket and connects to PC parallel port via modular telephone cable

Accepts 32K x 8 SRAM or NV SRAM

Loads Intel Hex, Motorola S, hex, and binary files

High and low RESET outputs for automatic startup after downloading

Includes all necessary cables and software -works right away!

PARALLAX

(916) 721-8217

FAX: (916) 726-1905

Parallax, Inc.
6200 Desimone Lane, #69A
Citrus Heights, CA 95621



California residents add sales tax.
Shipping: No charge for UPS ground,
\$9.00 for UPS 2nd day, \$18.00 for UPS next day.

Reader Service # 171

October/November 1990 77

you that the firmware to drive this thing is more complicated than the hardware...and that's no lie.

The NMC9346 uses a very specific protocol to pass commands, addresses, and data into the chip and read data and status back out. The SK (Serial Clock) input controls all of the timing, so, as long as you don't exceed the maximum clock frequency, there are no hard-and-fast timing restrictions. Each input or output bit is accompanied by an SK transition, so the firmware must read or write data relative to its SK output.

Writing data into the EEPROM requires an "Erase Register" to clear the previous data, then a "Write Register" command with 16 data bits. Reading data is similar: send a "Read Register" command, then clock 16 data bits out of the EEPROM. The EEPROM chip handles all of the timing required during erasing and writing, and the DO line goes high when the chip is ready for the next command.

Listing 3 shows what's required to send a single command to the

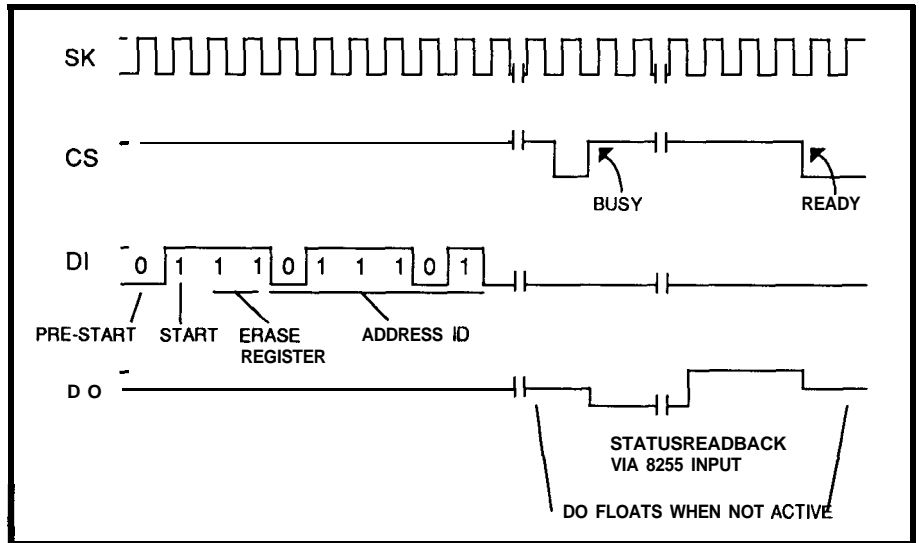


Figure 4-The signals shown are for 'Erase Register' with the register number 1D hex encoded in the lower six bits of the command byte. Notice that the DO line floats unless it is actually driving data from the NMC9346; the "Ready" status remains high until the CS line drops.

EEPROM chip. The signals shown in Figure 4 are for "Erase Register" with the register number 1D hex encoded in the lower six bits of the command byte. Notice that the DO line floats unless it is actually driving data from the NMC9346; the "Ready" status remains high until the CS line drops.

The NMC9346 is rated for 10,000 erase/write cycles per register, which means you cannot update the registers at "computer speeds" for very long. The EEPROM demo program requires you to press a key each time it writes into the chip so that it doesn't wear the poor thing out before you use it. I plan to update the EEPROM data twice a day, for an estimated life of about 14 years.

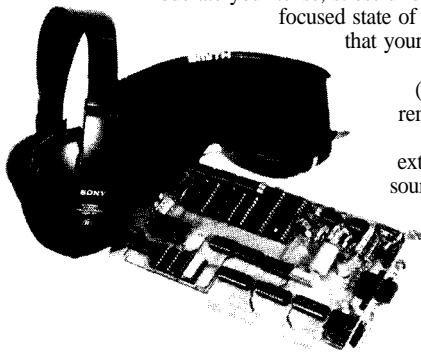
The EEPROM data should include a checksum to ensure that the data was written and read correctly. The Furnace Firmware code will write all 64 registers each time, simply because I'll have nearly 128 bytes of data. For you performance freaks, writing all 64 entries in one shot takes about a second.

There are several other types of NVRAM available; I just got a data sheet and sample from Dallas Semiconductor describing their DS2222 256-bit EconoRAM. It looks for all the world like a TO-92 plastic transistor: power, ground, and a single pin to multiplex control, clock, and bidirectional data into and out of the chip! **Not bad if you only need 32 bytes, is it?**

The demo program CDEMONVR.HEX creates a pseudorandom test pattern, writes it into the NMC9346, reads all 64 registers back, and compares the results. Any mismatches are identified by register number with the actual and expected data values. A

Convert your PC into a Brainwave Synthesizer.

By now you have probably read articles, or seen ads for mind machines... devices that moderate your tense, stressful brainwaves to produce a more relaxed, clearly focused state of mind. Most machines operate on the premise that your state of mind can be correlated to brainwave



MINDSEYE
SYNERGIZER

SYNERGIZER \$395
board, lights, software, manual

EXTERNAL CONTROL UNIT \$95
Software-assignable controls

HEADPHONES \$35

ORDER PHONE
800-388-6345
Credit Cards Accepted

© 1990

formation (206) 632-1722 SYNETIC SYSTEMS, INC PO Box 95530 Seattle, Wa 98145

frequencies, which are measured by EEG (electroencephalogram) patterns. While much remains speculative, what is known is that EEG brainwave patterns tend to "lock on" to an externally controlled source of flashing lights or sounds, and that the frequency of the brainwaves will follow the frequency of the external stimuli. *The implication is that you can influence your state of mind electronically.*

The MindsEye Synergizer™ is a powerful hardware/software combination that allows you to program and experience these shifts of awareness. It turns your IBM PC-XT, AT/386 or clone into a laboratory grade brainwave synthesizer. Synergizer™ sessions may be of almost any length and complexity, with each eye and ear programmed independently if desired; pulses can shift from one rate to another, while different sound frequencies are channeled left and right. Multiple time ramps and sound and light levels may be included within a single programmed session. A stereo synthesizer makes available a variety of waveforms, filters and other sound parameters. The Synergizer™ provides more programmable capabilities than any other available device at a remarkably low price.

Requires DOS 3.0 or above; 5 1/2 K of RAM.
and a hard drive are recommended.

Reader Service #2

```

;-----
; Serial console interrupt handler

_ConIntHandler PROC
    PUBLIC _ConIntHandler

    PUSH    ACC
    PUSH    PSW
    PUSH    DPH
    PUSH    DPL

;--- receiver interrupt

    JNB     RI,L?trans ; receiver interrupt active?
    MOV     A,RRing_cLevel; room for newest char?
    CJAE    A,#RINGSIZE,L?recdone ; no, so discard it

    INC     RRing_cLevel ; tick size counter
    MOV     A,RRing_cLevel; track maximum value
    CJBE    A,RRing_cMaxLev,L?1
    MOV     RRing_cMaxLev,A

L?1      MOV     DPTR,#RRing ; point to ring entry
    AddDPTR RRing_iHead
    MOV     A,SBUF ; pick up the character
    MOVX    @DPTR,A ; and stick in ring

    INC     RRing_iHead ; tick the index
    MOV     A,RRing_iHead ; hit end of array yet?
    CJB     A,#RINGSIZE,L?recdone
    MOV     RRing_iHead,#0; yes, reset to start

L?recdone CLR    RI ; indicate we've got it

;--- transmitter interrupt

L?trans  JNB     TI,L?done
    MOV     A,TRing_cLevel; anything to send?
    JNZ     L?2
    SETB    TransIdle ; no, flag inactive
    SJMP    L?transdone

L?2      CLR     TransIdle ; we're working on something
    MOV     DPTR,#TRing ; fetch next char
    AddDPTR TRing_iTail
    MOVX    A,@DPTR
    MOV     SBUF,A

    DEC     TRing_cLevel ; tick size counter
    INC     TRing_iTail ; tick the index
    MOV     A,TRing_iTail ; hit end of array yet?
    CJB     A,#RINGSIZE,L?transdone
    MOV     TRing_iTail,#0; yes, reset to start

L?transdone CLR    TI ; indicate we've done it

;--- clean up & go home

L?done   POP     DPL ; restore bystanders
    POP     DPH
    POP     PSW
    POP     ACC
    RETI

_ConIntHandler ENDPROC

```

listing 4—The serial port hardware causes an interrupt whenever it receives a new input character or finishes sending an output character. This interrupt handler buffers incoming and outgoing characters in a pair of ring buffers. The main program receives characters from the input ring and sends characters to the output ring while the interrupt handler takes care of the detailed timing and bit twiddling.

different test patterns used each time, **SUPER SERIAL**

but, because the random number generator in the C library doesn't return numbers over 32,767 the highest order bit will not be exercised.

The serial routines in the previous columns used polled I/O because it was simple and easy to understand.

Courteous Service • Discount Prices • Fast Shipping

ALL ELECTRONICS

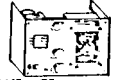
P.O. Box 567 • Van Nuys, CA 91408

12 Vdc 5 AMP POWER SUPPLY

ACDC Electronics# 12N5
or equiv. Input: 100-240 Vac
(wired for 115 Vac)

Output: 12 Vdc @ 5 amps.

Open frame power supply. 7" X 4 3/4" X 3" high.
Regulated. CATX PS-125 \$37.50 each



210 MFD 330 VOLT PHOTOFLASH CAPACITOR

Tubicon CE photoflash capacitor.

.79" dia. X 1.1" high. These are new

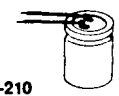
capacitors that have been prepped

with 1.4" black and red wire leads

soldered to the terminals. CAT#PPC-210

62.50 each. 10 for \$22.50 • 100 for \$200.00

Large quantities available. Call for pricing.



FLASHER LED

Diffused L.E.D. with built in flashing unit. PULSE
RATE: 3 Hz @ 5 Volt/20 ma. Unit continually flashes
when 5 Volts is applied. Operates between 4.5 Volts
and 5.5 Volts. T 1 3/4 size.

IDEAL FOR USE AS AN INDICATOR.

RED CAT# LED-4

GREEN CAT# LED-4G

YELLOW CAT# LED-4Y

\$1.00 each • 10 for 9.50 • 100 for \$90.00



REFLECTIVE OPTO SENSORS

These units have a IR emitter and sensor pair
pointing in the same direction. Light from emitter
bounces off object to be detected by sensor.
Effective range approx. 0.15.. Three types available:

TRW/ Optron# OPB5447-2
Rectangular w/ 28" wire leads
CAT# OSR-4 2 for \$1.00



TRW/ Optron# OPB703A
Wedge shape with PC pins.
CAT# OSR-3 75¢ each

TRW/ Optron# OPB711
Rectangular with PC pins.
CATX OSR-2 75¢ each

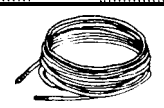


Optek# K-671 1
Wedge shaped device
CAT# OSR-1 75¢ each

RG 11/U 75 OHM VIDEO CABLE

Ivory RG 11/U terminated
with heavy-duty
F connectors.

Also includes a F-61
splice and a 75 ohm
terminator pad. New
cables manufactured for IBM pc
networks. UL listed. Specs: 75 ohm impedance. Cable
jacket O.D. 0.405. Dielectric 0.265" dia. Center con-
ductor 0.047" diameter. IBM PIN 1501906 COM/
SCOPE. CAT# RG-11-1100 foot roll \$15.00
CATW RG-11-2 200 foot roll \$27.50



TOLL FREE ORDER LINES

1-800-826-5432

CHARGE ORDERS to Visa, MasterCard or Discover

TERMS: Minimum order \$ 10.00. Shipping and handling
for the 48 continental U.S.A. 53.50 per order. All orders
including AK, HI, PR or Canada must pay full shipping.
All orders delivered in CALIFORNIA must include state
sales tax (6 1/4 %, 6 3/4 %, 7 1/4 %). Quantities Limited
NO C.O.D. Prices subject to change without notice.

Call Toll Free, or clip this coupon

FREE 60 Page Catalog
Containing over 4,000 ITEMS
ALL ELECTRONICS CORP.
P.O. Box 567 • Van Nuys, CA • 91408

Name

Address

City

State Zip

CC 10/90

Reader Service # 104

October/November 1990 79

However, polled I/O is inadequate for most applications because it can lose input characters. The solution is an interrupt-driven serial handler that responds immediately and buffers characters until the **program** can process them.

The serial interrupt handler routine is shown in Listing 4. The transmitter and receiver ring buffers are called `TRing` and `RRing`, naturally. Each buffer also has a set of associated variables: `cLevel` holds the number of characters in the buffer, `cMaxLev` records the highest value of `cLevel`, `iHead` is the index of the newest character, and `iTail` is the index of the oldest character. `RINGSIZE` sets the buffer size; the Furnace Firmware code uses 16 bytes for each ring.

A constant in EPROM sets the default serial rate to 19200 bits per second; in this application I don't need automatic rate sensing. The code does not implement XON/XOFF flow control, but adding that function is reasonably straightforward and is left as an exercise for the reader.

All of the "CDEMOxxx" programs use this serial driver, along with several interface routines that implement the C library functions `getch()`, `getche()`, `ungetch()`, `kbhit()`, `putch()`, `puts()`, and so forth. The demoprograms don't actually require much high-speed data handling, but it's nice to know it works.. .

MEET THE C MONSTER

At this point we have enough hardware to build a useful system: keypad, display, sensors, nonvolatile memory. All we lack is a program to pull it all together. Unlike Steve, however, I can't get away with programming in solder.. .

In principle, you should use the programming language most suited to the task at hand. For the 8031, you have several choices: assembler, assembler, and, if you don't like that, you can try assembler. If you spring for the 80C52-BASIC chip, the BASIC-52 language gives you a more-or-less familiar language that is reasonably

versatile, if not particularly convenient to use (reasonable men differ on this subject).

In recent years there have been several attempts to port C to the 8031 architecture. The early versions were, to put it charitably, lacking in performance, features, and what IBM called "your error-free use of the system" when it shipped a full refresh of OS/2 version 1.2. That seems to have changed, and the latest crop of C compilers are worth looking into.

To be fair, there are a number of other 8031 language systems available. Over the years, I've seen (or heard of) FORTHs, BASIC compilers, even a Pascal, and so on. However, if there is a mainstream language on 8031 CPUs, it's now C. Imitation being the sincerest form of flattery, there are even C-oid languages (Lite C?) which leave out the tricky (and useful) stuff like bit fields and structures.

The big motivation behind high-level languages is that you can get more done per line of code. Because you can only write and debug so many lines per day, it makes sense to use the most potent lines you can. The rule of thumb is that a single high-level language line is equivalent to 10 lines of assembler, so you might hope for an order of magnitude win.

Starting in late 1989, I beta tested the new Avocet 8051 C compiler, assembler, and simulator that form the heart of their AvCase product. Obviously, I'm not entirely objective about this, but I can extract some general observations from my experiences. Curt has an 8051 C compiler review in the works, so I'll save the "implementation details" for that.

You may think C is the solution to all your problems, that it allows you to get more work done faster, that it will produce "tight, hot code" with minimal effort, and that you can forget about assembler. My responses, in order, are: wrong, wrong, wrong, and dead wrong. Get the picture?

THE INCREDIBLE BULK

Based on measurements I've made on both Avocet's C library code and my own routines, each nonblank,

The DA/MTM

The Lowest Cost Data Acquisition System

Our DA/M can solve more of your data acquisition problems at a lower price than any other product on the market. DA/M's are used for:

- Military meteorological stations.
- Building management.
- Automated hydroponic farming.
- Industrial process control.
- Your application

In fact, DA/M's can be used whenever you can't afford to use any one else's product.

Made in North America, DA/Ms are available NOW!



Phone 1-403-486-3534

Fax 1-403-486-3535

Reader Service #130

noncomment line of C code compiled with the large memory model generates about ten 8031 machine instructions that occupy about 17 bytes of program memory. Optimizing for space can reduce that by about 10%, while optimizing for speed increases it by 15%. I haven't investigated other memory models because my code requires lots of data space.

Of course, other compilers will give different results. I suspect the results will be similar within a few percent, simply because code generation for the 8031 architecture is so difficult. If you're using (or thinking of using) a different compiler, make a few measurements yourself.

In any event, a typical 8031 system has either 8K or 32K bytes of EPROM, so your project must be doable in less than 450 or 1900 lines of C code. If you have big look-up tables or setup constants in EPROM, the code space decreases proportionately. If your lines of code are particularly potent, don't be surprised if each one requires more than 17 bytes!

I've recoded most of the standard C library routines in assembler to save space (and, of course, increase performance). One example will suffice: the `memcpy()` library routine to move a memory block was written in C, with "optimizations" to move words

rather than bytes if possible. The compiled code required 943 bytes.

I rewrote it in assembler, scrapped the word "optimization" because the 8051 can't handle more than a byte at a time anyway, and did a little tuning. The resulting code used 68 bytes. Then I realized it was quite similar to the `strcpy()` string move function and merged the two: the result handles both functions in the same 68 bytes.

Your mileage may vary, but when you find `printf()` adding nearly 4K bytes to your program, I predict you'll dust off your assembler skills in a hurry.

The C compiler defines a particular (also peculiar) subroutine calling protocol; a standard assembler routine won't work. I put "wrappers" around my existing code, leaving the original routines largely untouched. That way I can call the base routine from assembler or the new `entry` point from C with equivalent results. In most cases the wrapper code adds perhaps a dozen bytes, which is entirely OK.

Caveat: Avocet C passes the leftmost parameter in `R4:R5` and expects the return value in `R2:R3`. Other compilers have different conventions and, for each compiler, the conventions depend on the memory model. Take heed when you're porting these routines to a different environment!

RELEASE NOTES

The sample code for this column includes new C demo programs for the display and keyboard, as well as the ADC, clock chip, and the nonvolatile RAM. I'm also throwing in `CLIB.C51` and `CONSOLE.A51`, which have the assembler equivalents for some of the C library routines I've had to replace. Most of the driver code is useful for either assembler or C programs, but I've written some C-specific code for functions that I'll never call from assembler.

There have been a few bug fixes and some enhancements. One fix is of particular interest: the LCD initialization code used timing values "right from the data book" but I ran into an LCD board with a 150-kHz oscillator instead of the standard 250-kHz used in the specs. The new timing values suffice for LCD oscillators down to 100 kHz; check your board if the initialization seems erratic.

Compiling the demo programs has gotten a lot more complex; the `MAKEFILE.MAK` this time around has all the command line switches for the Avocet programs. One minor glitch is that I have more files than the C compiler driver can pass to the linker, so the link step force-feeds the linker everything the driver would normally extract from the Avocet configuration files. Take care if you're using different memory models or optimizations; I captured the driver output to disk and edited it to suit PolyMake's needs.

I'll wrap this project up next month with some buffered digital I/O and a program that may serve as the basis for your own data collector, sensor system, or what have you. Drop in on the BBS and tell me how your project is working out.

After that... well, stay tuned!+

Ed Nisley is a member of the Circuit Cellar INK engineering staff and enjoys making gizmos do strange and wonderful things. He is, by turns, a beekeeper, bicyclist, Registered Professional Engineer, and amateur raconteur.

IRS

268 Very Useful
269 Moderately Useful
270 Not Useful

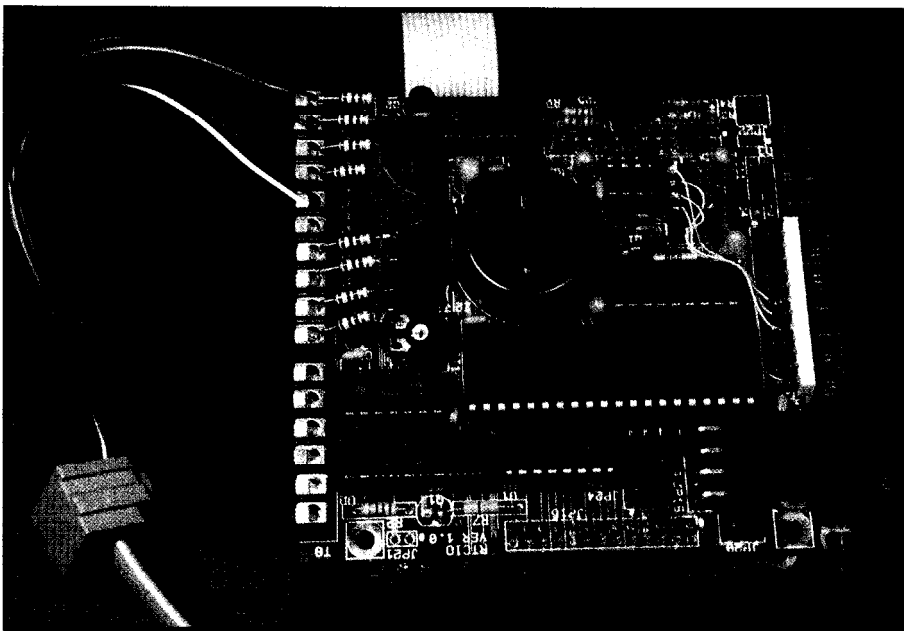


Photo 2-1 used hot-melt glue to mount the NMC9346 upside down in the spot normally used by the inductors for the DC-to-DC converter on the RTCIO board.

FROM THE BENCH

Jeff Bachiochi

PC Programming Comes to Embedded Control

V25...An 8088 with all the good stuff

The family of computers inspired by the IBM PC has very nearly become the universal standard platform for microcomputer development. From schematic capture and PCB layout, to compilers, assemblers and simulators, tools are available for the PC family that are unmatched on any other platform. Unfortunately for the embedded control developer, if your end product is a microcontroller, chances are it is not of the 80x86 family. This seems a shame, since many feel very comfortable with the PC family of processors. Unfortunately, the 80x86 series of processors require a good deal of support to make a total system since they were not designed with microcontrolling in mind.

PC clones are frightfully inexpensive, but their complexity and large size make them no match for a small microcontroller dedicated to a specific task.

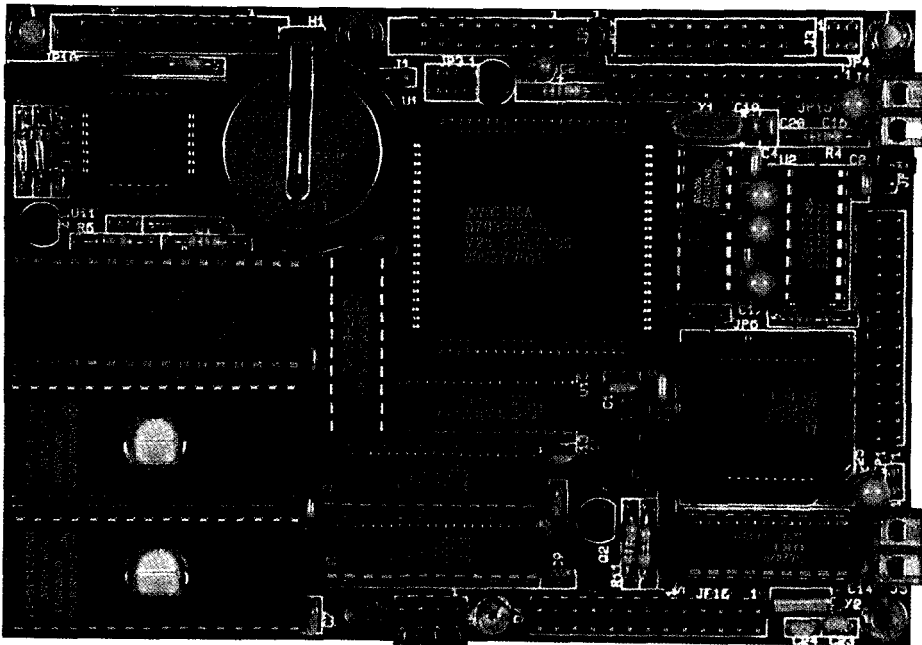


Photo 1 -The RTCV25 board sports a V25 processor, two serial ports, 40 parallel I/O lines, 8-channel 8-/10-bit ADC, up to 384K of RAM/EPROM, 128 bytes EEPROM, a battery-backed clock/calendar, and RTC stacking expansion connectors all on a 3.5" x 5" card (photo is actual size).

Why can't we have the power of PC but without all the fluff that we associate with our desktop? It's not as though I'm the first to ask this question: I've mentioned Intel's "WILDCARD-88" before. It's the small 8088 PC/XT clone on a credit-card-sized board. Built like SIMM, the 2- by 4-inch board was meant as the engine for your own RAM and I/O, which could be designed on separate "SIMM-type" boards and all bused together on a passive backplane. It's a nice modular design but it lacks many of the bit manipulation functions found in today's microcontrollers.

A few issues ago I unveiled the complete AT motherboard (including RAM) on a half-size PC expansion card, featuring Mitsumi's 286 Engine. The Mitsumi plug-in module which measures only 3 by 4 inches, con-

tains an amazing amount of silicon. It, too, must have the support of I/O devices and lacks good bit manipulation. It's an impressive step forward in miniaturization, but kind of overkill to replace a simple microcontroller.

PLAYING WITH PERFECTION

Back when 4.77-MHz PCs were all that could be found, we used to replace the Intel processor with an NEC V20. The innovative processor design increased throughput in some instances by more than 10%. (Hey! At that time we took it any way we could get it.)

NEC took this a step further with a "V" series offshoot. The V25 contains the high-performance V20 (software compatible with 8088/8086) along with 256 bytes of internal RAM, timers, DMA controller and serial/parallel I/O. (All this makes the V25 a microcomputer, which is a microprocessor with I/O, as opposed to the 8088 which is strictly a microprocessor.) The internal RAM is mapped as eight completely separate register sets, which make for some interesting possibilities. See Figure 1 for a V25 block diagram and Figure 2 for a memory map.

The V25 instruction set is a superset of the 8088/8086. Although fully code compatible with the 8088/8086, some execution times are slightly faster. If you use NEC's assembler, the mnemonic syntax is a bit different. In addition to the 8088/8086 instruction set, the V25 has some unique instruction enhancements. The most valuable instructions deal with bit manipulation. The ability to test, set, clear, or complement a single bit in a register or memory are the ultimate in control features.

BEGGING TO BE USED

When you happen across a processor such as the V25, little voices can be heard urging "Use me, use me." By the number of letters we've received on the RTC52, introduced here a year and a half ago, tiny microcontrollers are a popular commodity. This month I present a small-footprint, single-board (RTC bus-compatible) microcontroller

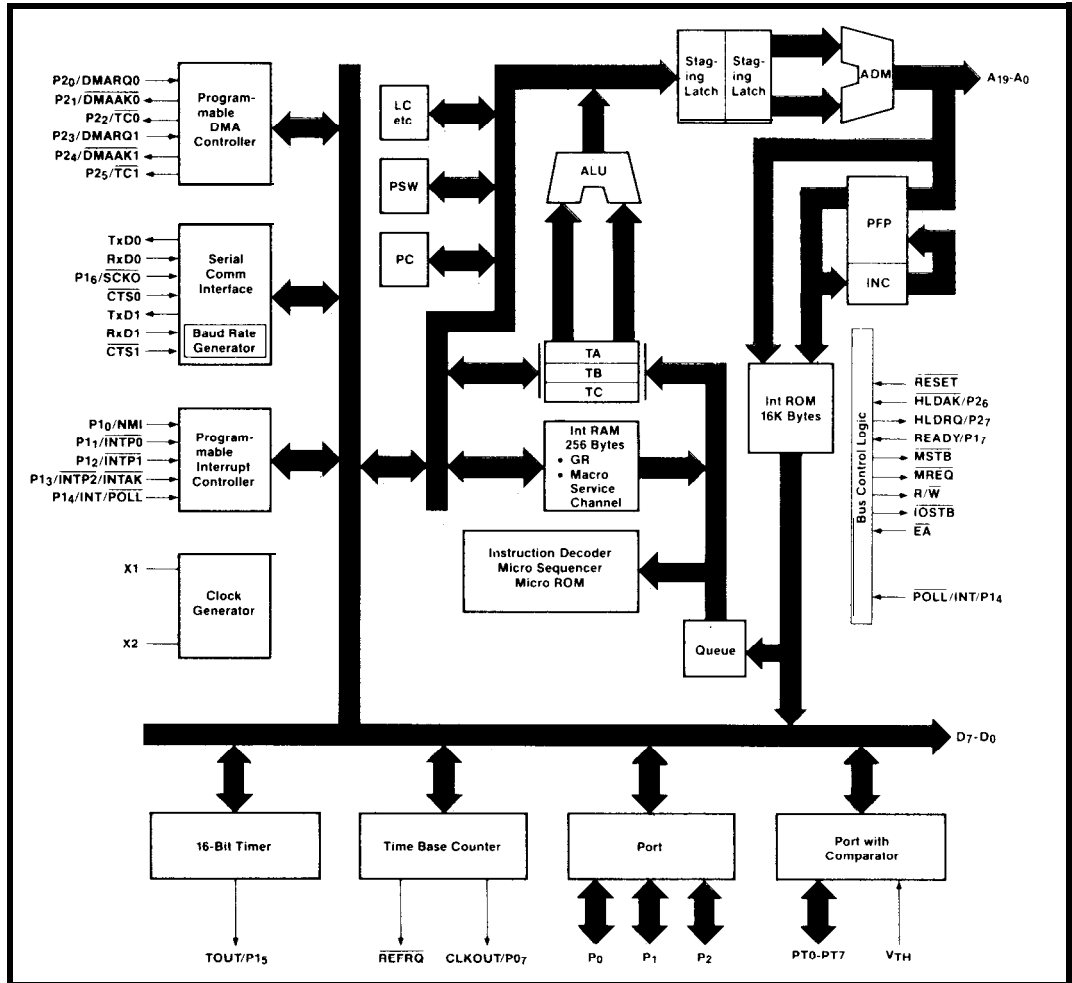


Figure 1 — The V25 chip includes everything found in an 8088 plus memory and an array of I/O devices.

using the NEC V25. See Photo 1 for a picture, Figures 3a-3c for the full schematic, and Figure 4 for a block diagram of the V25 board.

Let's start out by eliminating any rumors before they are started: This is an embedded controller and not a desktop computer! It will not be a PC-DOS-compatible machine. There is no BIOS support for high-resolution color graphics and no disks (floppy or hard). It is simply and purely an 8088/8086 code-compatible all-in-one workhorse.

THE BIG PICTURE

We are talking lots of addressable memory space on a V25 (1 megabyte) as compared to most microcontrollers (64K bytes), so 8K chips just won't do. To allow for the best combinations of RAM and ROM, without using up a "gob" of real estate, static RAM was chosen over dynamic. Two size options are available for memory: 32K- or 128K-byte devices. This allows the board to contain as little as 64K RAM/EPROM or as much as 384K RAM/EPROM. Thirty-two-kilobyte EPROMs are now in the \$10 price range, with 32K RAMs running about \$20. The price should continue to improve with the 128K devices falling into the same range within a few years.

Since control of the three memory sockets falls upon the system's PALS, intermediate sizes could be handled by altering the PALS' fuse maps. Type and size configuration is controlled by jumper headers connected to the PAL inputs. Each of the three memory sockets has a corresponding size selection jumper. The second memory socket has an additional type selection jumper. In the current design, RAM must populate the first socket which starts at address 00000H. EPROM must populate the third socket, which ends at FFFFFH and picks up the power-up program execution address of FFFFOH. The second socket is user selectable for either 32K or 128K RAM or EPROM. Mixed sizes are allowed for either device type to ensure maximum flexibility of the address area.

The V25 has built-in dual asynchronous UARTs complete with independent baud rate generators (110-38,400 bps). The hardware interface for one RS-232 port (with handshaking) and one RS-485 serial port or two RS-232 ports (without handshaking) is provided using a MAX232 and a TI SN75176. Square-pin headers are used for RS-232 and screw terminals for RS-485 to link the on-board communications ports with external equipment. Using the MAX232 provides RS-232 level conversion from a 5-volt-only source. The MAX232 needs four external capacitors (used as charge pump carriers).

Nonvolatile data storage can be useful in many applications. Small (in size and price) EEPROMs deliver in excess of ten thousand write operations making it ideal for semipermanent storage. The design includes a 1K EEPROM, which adds configuration storage capability to the V25 board. The NMC9346 serial EEPROM has 64 16-bit

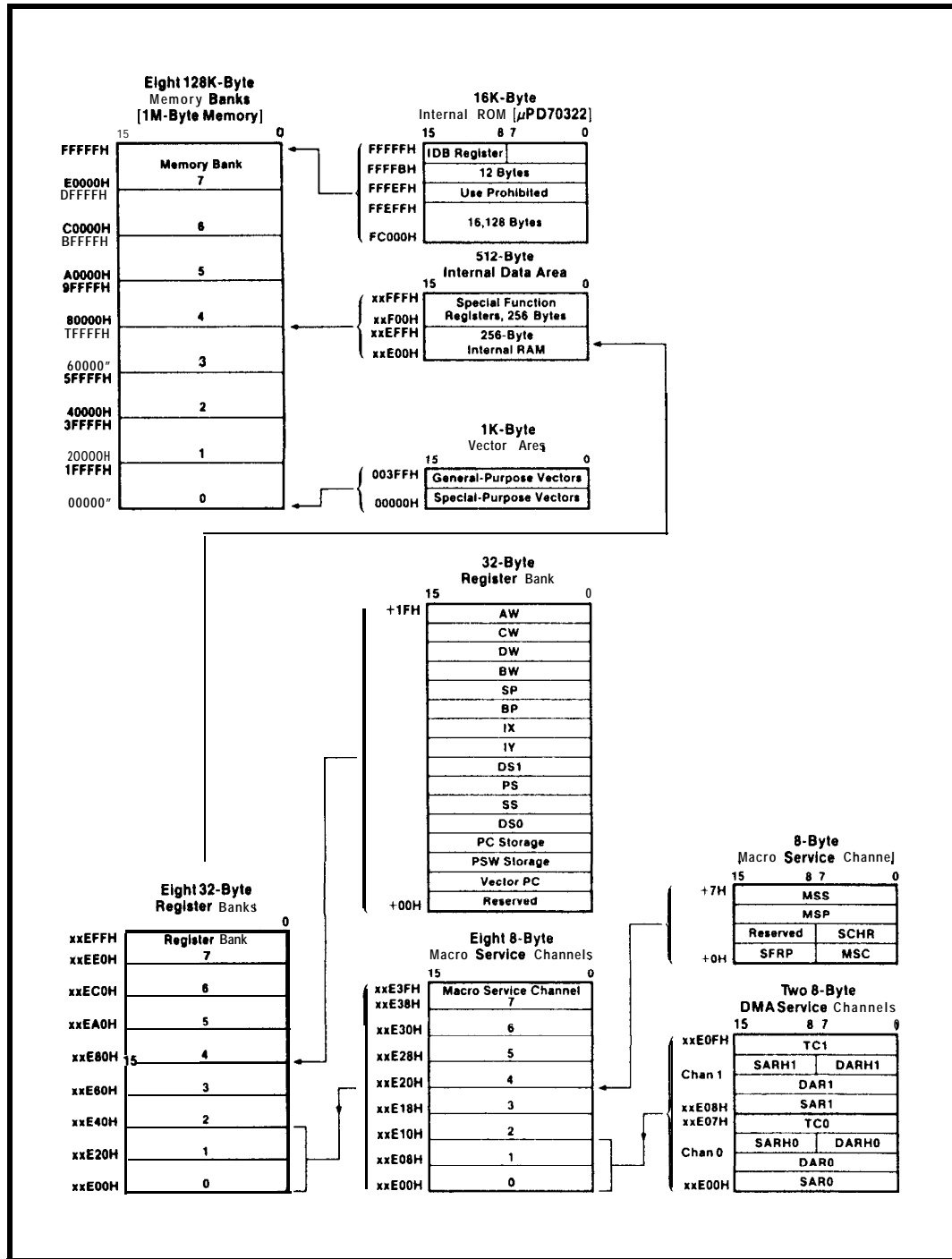


Figure 2—The V25's internal memory makes this processor more powerful than the standard 8088/8086

registers. These registers give the user a means of reprogramming a configuration change such as ID numbers or security access. [Editor's Note: For more details on how to use the NMC9346, see "Firmware Furnace" on page 73 of this issue.]

An Oki M6242B clock/calendar chip is included on the board which presents battery-backed time/date information to the system. Without the clock/calendar chip, it would be necessary to tie up one of the internal timers plus add the overhead of an interrupt service routine to keep track of time-of-day information. The clock's interrupt output can be used to wake up the processor from a HALT

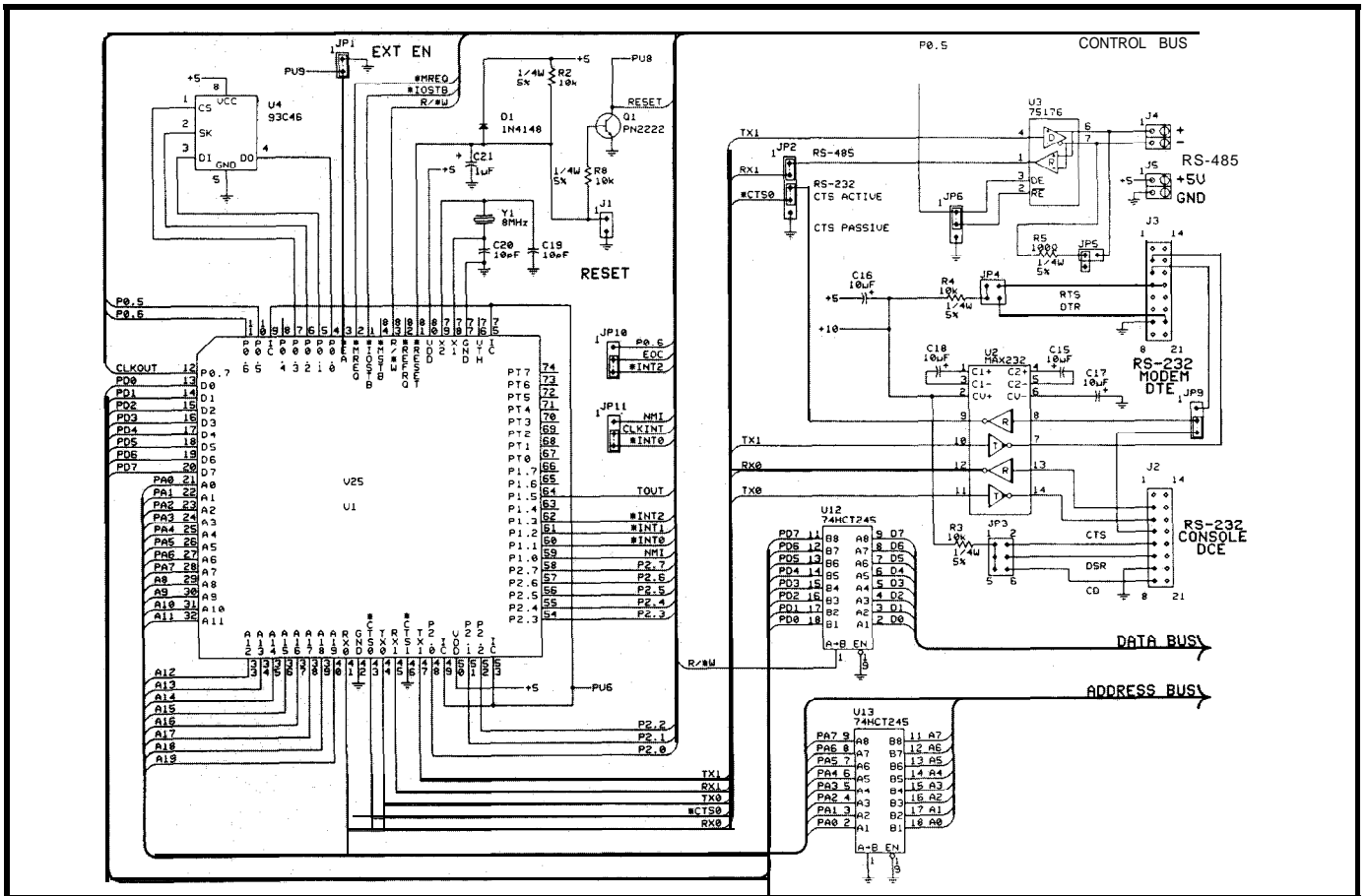


Figure 3a—The V25 includes virtually everything necessary to make a complete microcomputer in a single chip.

EXPRESS CIRCUITS

MANUFACTURERS OF PROTOTYPE PRINTED CIRCUITS FROM YOUR CAD DESIGNS
TURN AROUND TIMES AVAILABLE FROM 24 HRS — 2 WEEKS

Special Support For:

- TANGO.PCB
- TANGO SERIES II
- TANGO PLUS
- PROTEL AUTOTRAX
- PROTEL EASYTRAX
- smARTWORK
- HiWIRE-Plus
- EE DESIGNER I
- EE DESIGNER III
- PADS - PCB
- OTHER PACKAGES ARE NOW BEING ADDED
- FULL TIME MODEM
- GERBER PHOTO PLOTTING

Express
Circuits

314 Cothren St., PO. Box 58
Wilkesboro, NC 28697

Quotes:
1-800-426-5396
Phone: (919) 667-2100
Fax: (919) 667-0487

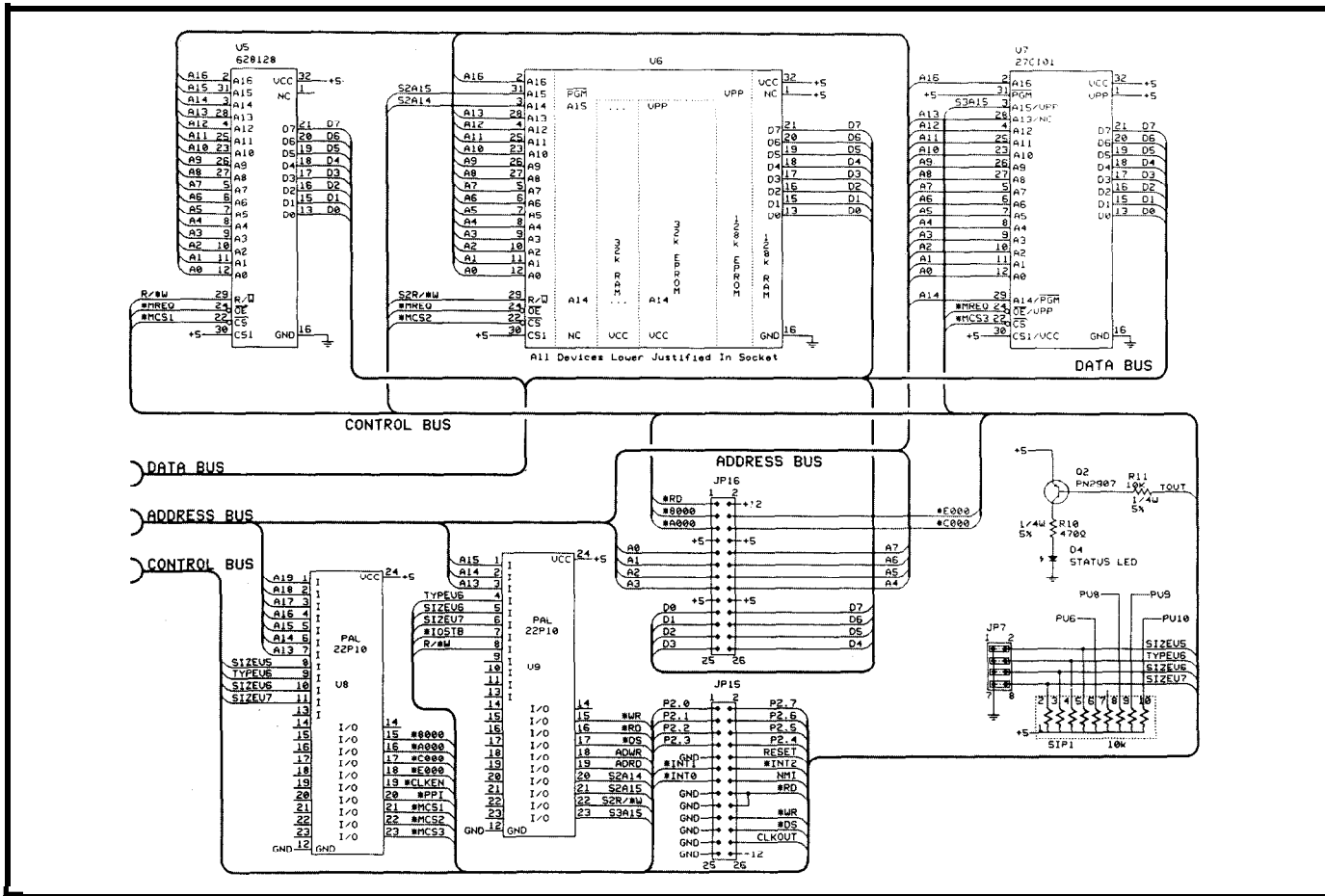


Figure 3b—A pair of PALs simplifies the design, makes a more compact board, and maximizes the number of memory device combinations possible.

or STOP mode at predefined intervals (1/64 second, 1 second, 1 minute, or 1 hour). This allows the user to take advantage of the lower current consumption provided by the HALT and STOP modes.

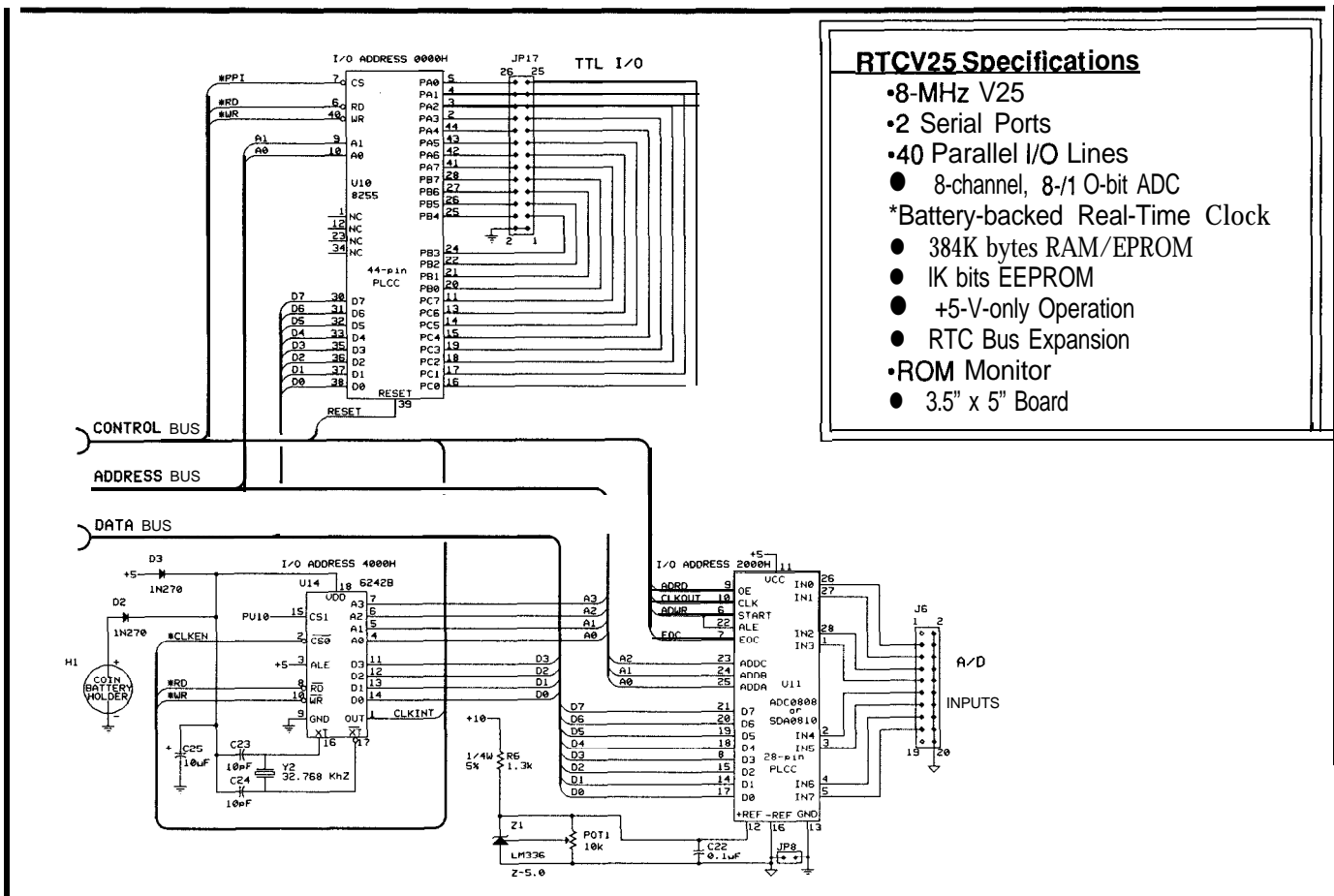
Although the V25 has 24 bits of parallel I/O, many of the pins have alternate functions and therefore cannot be used as system I/O. An 8255 PPI (Programmable Peripheral Interface) was added to fill this gap and keep the I/O structure similar to that used on other boards. Programming the mode (configuration) port of the 8255 can define each of the three 8-bit ports as input or output. These three parallel ports are brought out to a 26-pin header with the same pinout as the PPI's header on the RTCIO board.

The V25 also has an on-board 8-channel comparator with programmable threshold level, which is a fancy way of saying "a pokey 4-bit ADC." Using 5 volts as a full-scale reference, this 4-bit comparator equals about 300 mV/step. Since it is only a comparator, up to four comparisons would have to be made, changing the threshold level each time, to zero in on the actual value. Since 300 mV is not enough for most analog applications, I eliminated support for this on-chip function and substituted an ADC0808/ (SDA0810) 8- or 10-bit ADC (same as used on the RTCIO board) to yield a resolution of about 20 (or 5) mV/step. The 5-volt reference is generated by a reference diode from the +9-volt MAX232 output.

The 64K I/O address space (in addition to the 1M memory space) is divided into two areas. The area below 8000H is used for all on-board I/O including, the real-time clock, the PPI, and the ADC. The four 8K blocks above 8000H are decoded to provide I/O expansion through the vertical stacking bus introduced in the original RTC52 article. It only makes sense to keep any I/O you may have already designed for use with the other RTC processor boards compatible with this new V25 board. All of the signals which were on the RTC31/52 vertical expansion headers are reproduced as close as possible to the original signals. Even though the RTCV25 is designed for stand-alone operation, the I/O can be expanded through these vertical headers.

TEN POUNDS STUFFED INTO A 5-POUND BAG

Although the original RTC52 was about 3.5 by 3.5 inches, the I/O added to the board increased the size to 3.5 by 5 inches. The ability to be a useful stand-alone board makes the extra inch-and-a-half palatable and the 5-volt-only operation simplifies power supply requirements. I chose to use PLCC versions of the 8255 and the ADC0808/ SDA0810 to fit everything into the smallest size package. [Editor's Note: There are also DIP versions of the above chips available which have different pinouts and are physically larger



- ### RTCV25 Specifications
- 8-MHz V25
 - 2 Serial Ports
 - 40 Parallel I/O Lines
 - 8-channel, 8-1/2-bit ADC
 - * Battery-backed Real-Time Clock
 - 384K bytes RAM/EPROM
 - 1K bits EEPROM
 - +5-V-only Operation
 - RTC Bus Expansion
 - ROM Monitor
 - 3.5" x 5" Board

Figure 3c—To save space, the PLCC versions of the 8255 PPI and ADC0808/SDA0810 ADC chips are used.

than the PLCC versions. The schematics in this article assume the PLCC pinouts. Use care if you construct your own RTCV25 using DIP packages.]

Assemblers can be used to produce code for the V25 board using 8088/8086 syntax. These will not take advantage of the enhanced instruction set however, unless macros are used to support the extra functions. NEC does offer a Relocatable Assembler which includes the enhanced instructions. The syntax for the NEC instruction set is a bit different than that of the 8088/8086, somewhat like a Yankee talking to someone with a southern drawl.

A monitor ROM sets the foundation for a blitz of higher-level language interfaces. Monitor commands include:

- D (ump) memory block-in the usual l&byte hex and ASCII format
- E (nter) memory-display/change location value (forward and reverse address increments)
- F (i 11) memory block-with constant value
- G (o) -begin execution w/optional breakpoints
- H (elp) -displays these commands
- I (nput) -displays the value at a specific I/O address
- L (oad) -read in an Intel hex file from the host

ELECTRONICS

1 2 3
A DIVISION OF MING E&P, INC



5 SECOND EPROM ERASER
Revolutionary product
super energy output
Saves time & money
The most desired product
Patented design
MING IEE9088 \$249.99

1. Exclusive items at good price.
2. Unique items at better price.
3. Popular items at the best price.



ROM/SDRAM DISK CARD
For diskless PC station
Load DOS & file instantly
Battery back-up for SDRAM
Watch-dog timer rebooting
RD512 (512KB, 0KB) \$179.99
RD1024 (1024KB, 0KB) \$199.99



RF REMOTE CONTROL SYSTEM
19683 digital coding
2 tiny transmitters
Dry contact relay output
ON/OFF confirming signal
FCC approved
ZEMCO SA432 \$49.99

1-(800) -669-4406
TOLL FREE ORDER LINE

977 S. Meridian Ave., Alhambra, CA 91803
Tel: (818) 281-4066 Fax: (818) 576-8748
VISA & MASTER CARD ACCEPTED

- M**(ove) memory block-from/to block copy
- O**(utput) -writes a value to a specific I/O address
- R**(egister) --display/change a CPU register
- S**(FR) -display/change special function registers
- T**(race) -single step through program code
- U**(nassemble) -disassemble program code

The Circuit Cellar BBS has an area in the files section for "microprocessor cross-development tools." This area will be opened up to include the V25. Users are encouraged to retrieve and submit code for the V25. The monitor and PAL files are located there for your convenience. Also provided is NEC's macro library for supporting the V25's enhanced instructions. These macros are for use with your Intel ASM86 assembler, so you can take full advantage of the V25's **enhanced instructions**. [Editor's Note: See page 108 for information on downloading files from the Circuit Cellar BBS.]

In addition to the tools mentioned above, there are numerous commercial tools on the market that allow the programmer to use his favorite high-level language (usually C these days) to develop programs right on an IBM PC-compatible, then convert the final code (and add the necessary front-end initialization code) so it can easily be

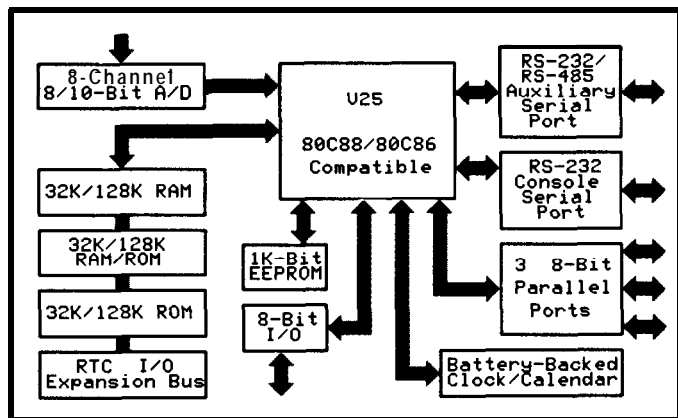


Figure 4—The RTCV25 board has more on-board memory and I/O than most embedded controllers.

placed in EPROM on an embedded controller like the RTCV25.

I dislike pumping out a design with no real benefits to anyone. That wasn't a problem here; the PC has deep roots as we enter the '90s. The processing power of the 8088 family is still acknowledged by industry and the comfortable feeling of writing code in a familiar environment will ease many into the microcontroller market without having to stray from their "native language."

Jeff Buchiuchi (pronounced "BAH-key-AH-key") is a member of the Circuit Cellar INK engineering staff. His background includes work in both the electronic engineering and manufacturing fields. In his spare time, Jeff enjoys his family, windsurfing, and pizza.

\$249. TERMINAL

Featuring

- Standard RS-232 Serial Asynchronous ASCII Communications
- 48 Character LCD Display (2 Lines of 24 each)
- 24 Key Membrane Keyboard with embossed graphics.
- Ten key numeric array plus 8 programmable function keys.
- Four-wire multidrop protocol mode.
- Keyboard selectable SET-UP features—baud rates, parity, etc.
- Size (5.625" W x 6.9" D x 1.75" H), Weight 1.25 lbs.
- 5 x 7 Dot Matrix font with underline cursor
- Displays 96 Character ASCII Set (upper and lower case)

Options—backlighting for display, R-422 I/O, 20 mA current loop I/O,

COMPUTERWISE, INC.

302 N. Winchester • Olathe, KS 66062 • 913-829-0600 • 800-255-3739

SOURCES

Chips

Circuit Cellar Kits
 4 Park St., Suite 12
 Vernon, CT 06066
 (203) 875-2751

V25 monitor in a 27C256 \$50.00
 U8 and U9 programmed PALs \$30.00/pair
 Please add \$3 shipping and handling in u.s.; \$8 elsewhere.

RTCV25

Assembled and tested RTCV25 boards are available from Micromint Inc., (203) 871-6170. Call for more information.

μPD70320/322 (V25) User's Manual

NEC Electronics, Inc.
 401 Ellis St.
 P.O. Box 7241
 Mountain View, CA 94039
 (800) 6323531 (for literature)
 (415) 960-6000

IRS

- 271 Very Useful
- 272 Moderately Useful
- 273 Not Useful

VHDL— The End Of Hardware?

SILICON UPDATE

Tom Cantrell

Those following the writings of our illustrious leader, Steve Ciarcia, know he has some strong feelings about hardware and software. For Steve, software is a necessary evil required to make a collection of “iron” (er, silicon) do its stuff. Of course, the explosion of low-cost microprocessors has forced him to accept programming to some degree, but I get the impression he’d rather face a root-canal than a coding session.

Meanwhile, recent political changes around the world have led some to proclaim “The End Of History.” The thought is that the various ideologies-whose conflicts have fueled major events over the centuries-are converging into a common quasidemocratic/semisocialist scheme.

You (and my friendly editor) are undoubtedly asking what root-canals and “The End Of History” have to do with the chips I’m supposed to be writing about. Well, recent events here in Silicon Valley indicate that the battle between software and hardware “ideologies” may be coming to a close. In particular, the emergence of “logic synthesis” and HDLs (Hardware Description Languages) marks the beginning of the end for hardware as we know it. For hardware purists, the transition may be painful (imagine serving as the “test-bed” for root-canal practice at the local dental college). However, those that resist the new order will be purged. Simply put-adapt or die!

SO MANY GATES, SO LITTLE TIME

Human nature is such that the way things are done is rarely changed until there is a “crisis” of some sort or another. Only then will a new order be imposed (best case, at the ballot box; worst case, in the streets).

Assuming most of you are chip users, not chip designers, the “crisis” driving the HDL revolution

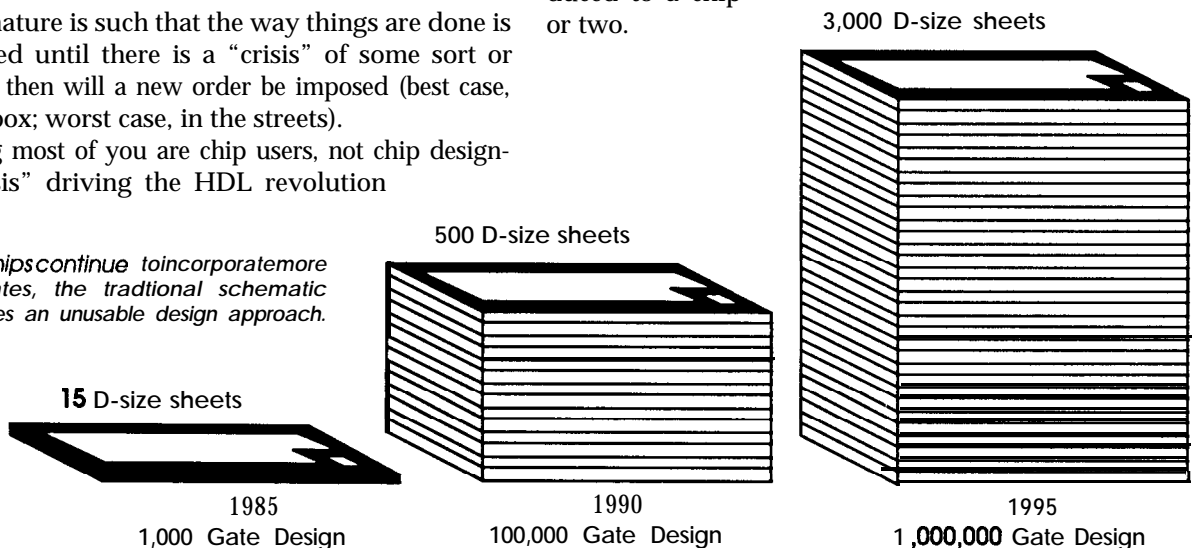
may not be apparent. After all, the traditional “schematic” approach to hardware design seems to work just fine. The plots for the typical **CIRCUIT CELLAR INK** project are rarely more than 1-3 pages long, certainly no reason for despair.

However, for the chip designer, the crisis in hardware design is much more apparent. Ironically, the chip-design crisis is the fault of the chip designers themselves. Specifically, the poor fools insist on making chips that are more complex, cost less, and are delivered to market as quickly as possible. Now, they’ve backed themselves in a corner.

The problem is that chip density and cost/time pressures are such that the “schematic” approach is arguably obsolete now, and will certainly be untenable in the near future. Today’s one-million-transistor chip is nothing compared to tomorrow’s silicon marvels. And “tomorrow” doesn’t mean some 20-30 years from now when you are retired. I hear 64-megabit DRAM prototypes are already working in the lab while Intel has announced the i586 (2 million transistors), i686 (4-5 million transistors), and i786 (100 million transistors!) upgrade path for the nineties. Unfortunately, as Figure 1 shows, schematics just aren’t going to cut it for the next generation of **SuperChips**.

Actually, to the degree chip designers face and solve the crisis, chip users won’t have to. Assuming the chip designers are successful, system designers’ schematics will be simpler than ever, since everything you could ever want will be reduced to a chip or two.

Figure 1 — As chips continue to incorporate more and more gates, the traditional schematic quickly becomes an unusable design approach.



PALS 16R4

```

1.  PHI          11. /OE
2.  PHI          12.  A0
3.  /SET-TWO    13.  SYNC-INH
4.  CDSB        14.  /LATCH-DATA
5.  WAIT        15.  STB-ENA
6.  /AS         16.  /TWO-CYC
7.  /MAS        17.  /WAIT
8.  /UDS        18.  SYNC
9.  /LDS        19.  /DTACK-INH
   GND          20.  vcc

/ SYNC          = / CDSB (TRISTATE ENABLE)
+ /MAS
+ /UDS * /LDS
+ / SYNC * / SYNC-INH * / PHI

/ SYNC-INH     = HI (TRISTATE ENABLE)
+ /AS
+ / SYNC-INH * / SYNC
+ / /TWO-CYC * / STB-ENA * / PHI

/ /TWO-CYC     = SYNC * / /SET-TWO * / /AS
+ / /TWO-CYC * STB-ENA * / /AS

/ STB-ENA      = / SYNC * / STB-ENA
+ / SYNC * /WAIT

/ /WAIT        = WAIT

/ /LATCH-DATA  = / /TWO-CYC * /WAIT * / /AS
t / /LATCH-DATA * / /AS

/ /DTACK-INH   = HI (TRISTATE ENABLE)
+ /UDS * /LDS
t / /SET-TWO
t / /TWO-CYC

/ A0           = HI (TRISTATE ENABLE)
+ / SYNC-INH * / /UDS * /LATCH-DATA
+ SYNC * / /UDS * /LATCH-DATA

PAL.

```

```

0  ---- -X- -X- -X- -X- -X- -X- -X- -X-
1  ---- -X- -X- -X- -X- -X- -X- -X- -X-
2  ---- -X- -X- -X- -X- -X- -X- -X- -X-
3  ---- -X- -X- -X- -X- -X- -X- -X- -X-
4  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
5  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
6  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
7  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
8  ---- -X- -X- -X- -X- -X- -X- -X- -X-
9  ---- -X- -X- -X- -X- -X- -X- -X- -X-
10 ---- -X- -X- -X- -X- -X- -X- -X- -X-
11 -X- -X- -X- -X- -X- -X- -X- -X-
12 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
13 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
14 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
15 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
16 ---- -X- -X- -X- -X- -X- -X- -X- -X-
17 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
18 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
19 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
20 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
21 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
22 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
23 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
24 ---- -XX- -XX- -XX- -XX- -XX- -XX- -XX- -XX-
25 ---- -XX- -XX- -XX- -XX- -XX- -XX- -XX- -XX-
26 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
27 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
28 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
29 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
30 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
31 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
32 ---- -X- -X- -X- -X- -X- -X- -X- -X-
33 ---- -X- -X- -X- -X- -X- -X- -X- -X-
34 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
35 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
36 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
37 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
38 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
39 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
40 ---- -X- -X- -X- -X- -X- -X- -X- -X-
41 ---- -X- -X- -X- -X- -X- -X- -X- -X-
42 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
43 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
44 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
45 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
46 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
47 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

```

Figure 2—PALs can be used to replace numerous discrete logic chips and are defined using Boolean equations rather than schematic diagrams. At the right is the resulting fuse map used to burn the final PAL.

HDL-HARD SOFTWARE OR SOFT HARDWARE?

The trend towards “softer” hardware has been growing. One of the best examples is the PAL (Programmable Array Logic), which is often programmed using Boolean equations. These Boolean equations are in fact a kind of HDL.

You can’t wire up Boolean equations. The process of “translating” the “abstract” equations into real hardware (gates, or more correctly in the case of a PAL, wiring connections between predefined gates) is called “logic synthesis” (Figure 2).

Even a simple PAL design represents a fundamental shift in the hardware design paradigm from the traditional schematic approach to a programming-language scheme. After all, the Boolean language is called PALASM, not PALGATES. When schematics are finally dead, historians may note the emergence of the PAL was the first, albeit small, step on a very slippery slope.

One driving force for HDL has been the goal of the ASIC (Application-Specific Integrated Circuit) idea. A system designer can (hopefully) create his own chips which are then manufactured by a semiconductor manufacturer (also known as a “foundry”). Unfortunately, since even dedicated teams of chip designers are in crisis, the “casual” ASIC designer doesn’t have a chance. Designing chips using the traditional methods is only feasible for customers with **really deep** pockets. They should also have a high tolerance for pain and frustration since the ASICs rarely work without many marathon debug sessions. The promise of ASIC will never be fulfilled unless higher-level tools like HDL are provided.

Actually, leading-edge chip designers are already using HDLs; “synthesized” chips are at work today. These users report that the benefits of HDL are not illusory; progress in design productivity, flexibility, and time-to-market is real. Yet, so far the HDL scheme has suffered from market and technical immaturity. What exists is a rather ad hoc con-

glomeration of various HDLs, platforms, suppliers, libraries, foundries, and so on. The situation is kind of like the earliest stages of the personal computer market which didn't reach critical mass until a certain level of standardization was achieved.

What's been needed is an HDL equivalent of an IBM PC to bring order to the chaos.

VHDL

"VHDL" stands for "VHSIC Hardware Description Language" ("VHSIC" stands for "Very High Speed Integrated Circuit"-I guess we've finally entered the era of nested acronyms!).

Before getting into the technical details of VHDL, you should understand that they don't affect the outcome. VHDL is the winner. On the other hand, not much is truly available yet. I'm unaware of any VHDL-synthesized chips at this time. But that will change; the bandwagon is unstoppable.

VHDL was developed in the early '80s at the behest of the Department of Defense (DOD) by Intermetrics, TI, and (you guessed it) IBM.

Besides handling complexity, DOD has a unique problem when it comes to fast-changing, high technology. Design and procurement cycles are somewhat glacial and many times problems aren't discovered until long after the original work was started. In the worst case, the DOD may find that the original company, tools, and even human designers needed to build/fix something are all dead! The DOD wanted to devise "languages" that would allow long-term maintainability.

The government's solution for "mission-critical" software is Ada, and there is no avoiding the fact that VHDL shares many of the same concepts. Before you extrapolate the potential success of VHDL from that of Ada, remember that the fundamental market situation for each is quite different. Ada emerged as an "incremental" improvement of popular and mature commercial languages: C, FORTRAN, Pascal, and so on. The technical features of Ada may not be compelling in the face of the entrenched alternatives. On the other hand, VHDL entered

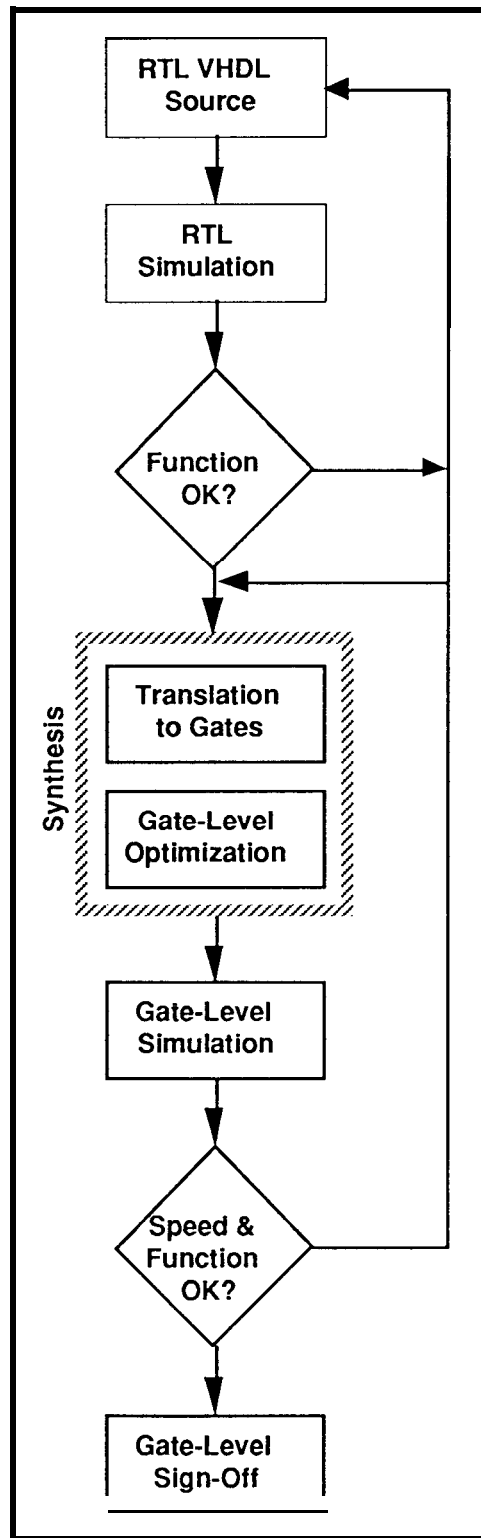


Figure 3-A *very structured design policy for VHDL design is recommended.*

the vacuum of a small, young, fragmented HDL market. Thus, the HDL war (more like a skirmish) is over almost before it started.

Since 1987 VHDL is an ANSI and IEEE standard—more momentum behind the language. If that's not enough (after all, the S-100 bus is an IEEE standard too), the real kicker is that DOD is starting to demand a VHDL description of chips they buy.

TYPING IN A CHIP

Actually, VHDL is kind of a universal programming language. As far as I can tell, it can do anything that C or Pascal can, and even more. In actual practice, chip designers are focusing on the "synthesizable subset" of VHDL. For instance, VHDL includes file I/O but it makes little sense to synthesize the chip equivalent of a PRINT statement.

One of the avid promoters of VHDL, an outfit called Synopsis, recommends the VHDL design policy shown in Figure 3.

The process starts with entry of "RTL VHDL" source code describing the chip. "RTL" refers to "Register Transfer Level," a method of description in which all storage elements and states in the network are explicitly defined.

Next, the RTL VHDL model is simulated to verify the chip's behavior. This level of simulation is concerned with the overall functional correctness of the design, not details of timing and loading. For instance, a latch can be shown to store data when strobed, but the setup/hold times and such are not proven yet.

After the high-level behavior of the chip is verified, logic synthesis automatically translates the high-level description to gates and optimizes the resulting net list.

One of the key goals of VHDL is "technology independence," that is, the high-level description can be freely targeted at a variety of processes. Remember, **when** DOD discovers twenty years after the fact that their bombs might explode accidentally they might want to rework a chip whose original technology may be long gone.


```

entity VHDL is
  port (
    A, B, C: in BIT;
    Z      : out BIT
  );
end VHDL

architecture VHDL_1 of VHDL is
begin
  Z <= (A and B) or C;
end VHDL_1

```

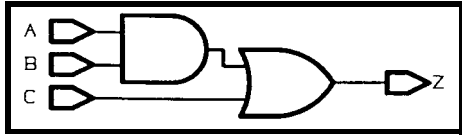


Figure 4-A very simple logic example serves to illustrate the basic entity/architecture approach of VHDL.

So, the final step is to reverify the design at the gate level. This time, the functional behavior of the chip presumably need not be checked (at least to the degree you trust the synthesizer). What's determinable at the gate level (and not determinable at the RTL level) is the actual timing and margins of the device, which ultimately depend on the foundry, process, and circuit library which will be used.

```

entity VHDL is
  port (
    A, B : in INTEGER range 0 to 15;
    C      : out BOOLEAN
  );
end VHDL;

architecture VHDL_1 is
begin
  C <= (A < B);
end VHDL_1

```

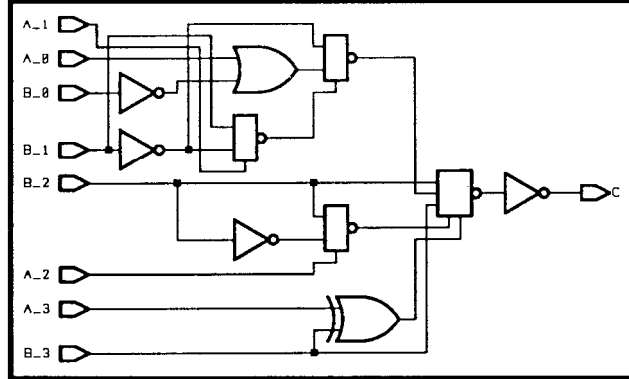


Figure 5—Without adding any complexity to the VHDL source file, many more gates are generated, showing the potential power of VHDL definition.

Professional text editor for only \$29.



- Pull-down menus, mouse support
- Columnar blocks, 1000 level Undo
- Edit 100+ megabytes file, super fast
- Also VEDIT \$69, VEDIT PLUS \$185

FREE Evaluation Copy - 1-800-45-VEDIT

The VEDIT family of text editors offers stunning performance, versatility and ease of use. Completely written in assembly language, they are small and lightning fast. Edit text and binary files of any size, even 100+ megabytes. Installation is trivial; VEDIT.EXE and an optional help file are all you need. Easily installs on a floppy disk.

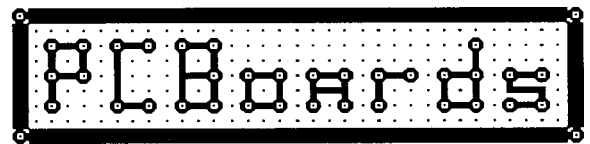
VEDIT Jr. includes pull-down menus with "hot keys", context sensitive help, pop-up ASCII table, 1000 level Undo, block operations by line, character, file or column, a configurable keyboard layout and keystroke macros. Automatic indent, block indent and parentheses matching. Word wrap, paragraph formatting, justification, centering, adjustable margins and printing. Run DOS programs. All for only \$29.

VEDIT can simultaneously edit up to 36 files and split the screen into windows. Search/replace with regular expressions. The new compiler support in VEDIT runs not only popular compilers, but debuggers and your favorite tools from within the editor. When shelling to DOS, VEDIT swaps itself and any desired TSRs out of memory. Only \$69.

VEDIT PLUS adds a powerful "off the cuff" macro programming language, complete with source level debugging. The macro language includes testing, branching, looping, user prompts, keyboard input, string and numeric variables, complete control over windows plus access to hardware interrupts, memory and I/O ports. Only \$185 for DOS, \$285 for UNIX/XENIX, QNX or FlexOS.

Greenview Data

P.O. Box 1586, Ann Arbor, MI 48106
(313) 996-1299 • Fax (313) 996-1308



P-C-B ARTWORK MADE EASY !

Create Printed Circuit Artwork on your
IBM or Compatible

- * MENU DRIVEN
- * HELP SCREENS
- * ADVANCED FEATURES
- * EXTREMELY USER FRIENDLY
- * AUTO GROUND PLANES
- * IX and 2X PRINTER ARTWORK
- * 1X HP LaserJet ARTWORK

* HP and HI PLOTTER DRIVER optional 49.00

REQUIREMENTS: IBM PC or Compatible, 384K RAM
DOS 3.0 or later. IBM compatible printers.

PCBoards - layout program **99.00**

PCRoute - auto-router **99.00**

SuperCAD - schematicpgm. **99.00**

DEMO PKG. - 10.00

Call or write for more information
PCBoards

2110 14th Ave. South, Birmingham, AL 35205
(205) 933-1122

```

entity VHDL is
  port (
    WORD    : in BIT-VECTOR (0 TO 7);
    PARITY  : out BIT
  );
end VHDL;

architecture VHDL_1 of VHDL is
begin
  process
    variable RESULT : bit;
  begin
    RESULT := '0';

    for 1 in 0 to 7 loop
      RESULT := RESULT xor WORD(1);
    end loop;

    PARITY <= RESULT;
  end process;
end VHDL_1;

```

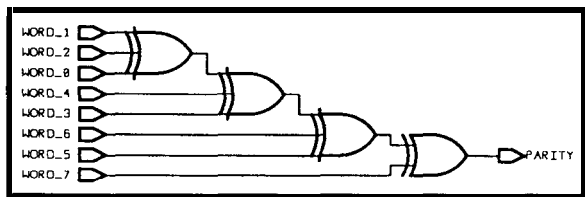


Figure 6-The VHDL 'for' statement may be used to iterate sections of logic.

Those of you with any software experience will recognize the similarity of this VHDL design scheme to the traditional edit/debug-compile/cycle. The synthesis of gates from RTL VHDL source is quite similar to the compilation of object code from source code. On further examination you will find that many VHDL concepts have direct analogs in the software world.

SOFT HARDWARE OR HARD SOFTWARE ?

Let's take a look at some actual VHDL code so you can get an idea of how it all works.

Figure 4, though a trivial bit of logic, illustrates the basic entity/architecture approach of VHDL. The entity portion defines the interface (i.e., inputs/outputs) to the synthesized hardware while the architecture determines the actual implementation. Software types will immediately note the similarity to the modem programming language practice of separating the definition and implementation of a function.

The benefit of VHDL seems questionable in this example. After all, entering the two-gate schematic is probably easier, and certainly less error prone, than typing in 10 lines of VHDL code.

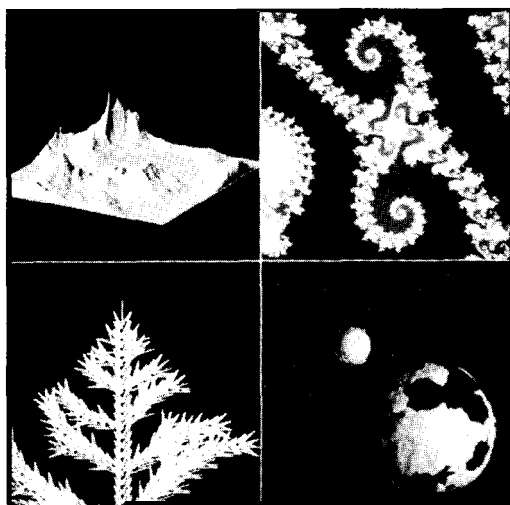
Figure 5 starts to show how VHDL can amplify design productivity. Note that this function (a 4-bit comparator) also requires 10 lines of code, but this time many more gates are generated. Furthermore, it's easy to see even

3-D GRAPHIC ART/IMAGING MULTI-DIMENSIONAL FRACTALS

in an image processing environment

FRACEDT is a 5-disk multi-purpose software package that combines the world of fractal imagery with that of image processing. The package is composed of four different fractal codes which create 1, 2, and 3-D images: such as fantastic designs in CHAOS, plants, trees, and curves using Lindenmayer systems; mountains, continental masses using Fractional Brownian motion and 3-D fractal planetary scenes. Five image processing utility codes which enable you to pseudo color, zoom, translate, rotate, and mathematically manipulate multiple images to create fantastic composite images. These utilities support a full 3-D projection mode with hidden surface, shading and lighting effects.

MOUNTAINS, LANDSCAPES CHAOS



PLANTS, TREES

PLANETS

Requirements:

PC/AT with an EGA or VGA, DOS 3.3 or later, 512K RAM and an HDD

Suggested retail price:

\$69.95 for total package with documentation or \$5.00 for two demo disks

Please include \$3.50 shipping and handling with each order.

TARDIS SYSTEMS

945 San Ildefonso, Suite 15 • Los Alamos, NM 87544 • (505) 662-9401

more potential. The function could be changed from a 4-bit to an 8-bit comparator simply by changing the "INTEGER range 0 to 15;" statement to "INTEGER range 0 to 255;"—no additional lines of code, but many more gates.

In a similar manner, VHDL supports "vectors" and "loops" which easily generate replicated logic as shown in Figure 6. Once again, the 8-bit parity generator shown could be easily changed to any size by simply changing the vector size and loop counter.

Though decoupling the designer from the details of gate-level design is a **worthy** goal, it must be tempered. Of notable concern is the traditional desire to tune a design by making tradeoffs between area and speed. To that end, VHDL includes "synthesis attributes" which allow the designer to convey the desired tradeoffs. For instance, as shown in Figure 7, a **single** VHDL definition may generate completely different gates depending on the attributes specified.

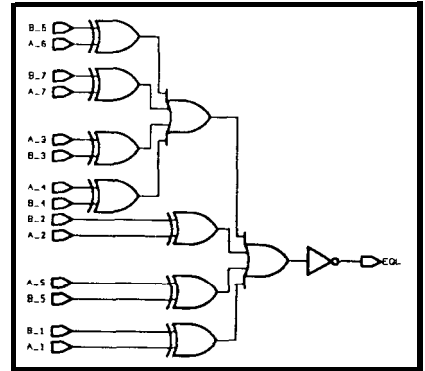
THE REAL WORLD

These examples just touch on the capabilities of VHDL. The language includes many high-level constructs like packages (libraries of predefined functions), enumerated (user-defined) types, function and operator overloading (kind of like OOP), and so on which all serve to support large, hierarchical designs. Just as a "Hello world" program seems to take a lot of lines/bytes of code, these small VHDL examples don't illustrate the efficiency achievable

a)

```
entity VHDL_SMALL is
  port (
    A, B : in BIT-VECTOR (1 to 7);
    EQL : out BOOLEAN
  );
end VHDL_SMALL;

architecture VHDL_SMALL_1 of VHDL_SMALL is
  attribute MAX_AREA of VHDL_SMALL--: entity is 0;
begin
  EQL <= (A = B);
end VHDL_SMALL_1;
```



b)

```
entity VHDL_FAST is
  port (
    A, B : in BIT-VECTOR (1 to 7);
    EQL : out BOOLEAN
  );
end VHDL_FAST;

architecture VHDL_FAST_1 of VHDL_FAST is
  attribute MAX_DELAY of EQL : signal is 0;
begin
  EQL <= (A = B);
end VHDL_FAST_1;
```

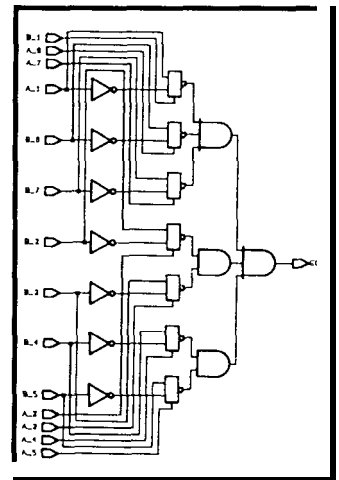
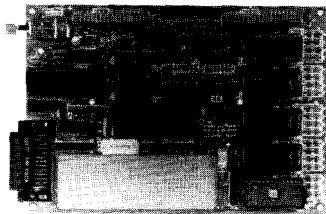


Figure 7-Attributes may be assigned to a section of logic such that if is optimized for (a) minimum area or (b) minimum delays. Note that the resulting logic is identical in both cases, but each is optimized for the attributes assigned to it.

New, Improved!

SIBEC-II

The ideal solution for embedded control applications and stand-alone development.



- Intel 8052AH BASK CPU
- Serial printer output and 5, 8 bit I/O ports
- 5 in.² prototyping area
- Memory: 8K RAM, expandable to 128K
- Power requirements: 5V.DC @ 300 ma. on/y
- PROM programmer; ZIF socket for 2764 or 27128 EPROM
- Interrupt handling capability
- Built to exacting standards and warranted
- Still only \$228.00 including documentation (quantity 1)

Inquire about our PDK51 80518052 product development kit for the IBM-PC/XT/AT: \$595.
Our BXC518051/8052 BASIC compiler: \$295.

Call now! 603-469-3232



Binary Technology, Inc.

Main Street . PO Box 67 . Meriden, NH 03770



Reader Service #113

in larger designs (which reduce the ratio of overhead to function). Like software programs, a real-world VHDL design will typically require hundreds if not thousands of lines of code and the bigger the design, the bigger the payoff.

Besides the direct design-dollar-per-gate advantage of VHDL, don't overlook other benefits that accrue.

Traditionally, chip-design teams are partitioned into architecture, circuit design, layout, simulation/test, and so forth. This leads to overhead and ball dropping. With VHDL, the team is unified around a single model right from the start.

The simulation orientation of VHDL allows testing to take place from the earliest stages of the design. This encourages lots of "what if" tweaking by the designers. With the traditional approach, a design team might get quite deep into a gate-level design before a fatal roadblock is encountered. With VHDL, such dead-end paths can be identified and avoided early on.

Technology independence not only allows shopping around for the best foundry but eases the migration to new processes as they emerge. Gone are the days when an old, but usable, design must be manually reworked because the original manufacturing technology is becoming obsolete. The VHDL design is simply "resynthesized" with the latest and greatest process as the target.

VHDL is poised to take off. If you're interested in chip design, now is the time to start checking VHDL out. Even

if you're only interested in using chips, not designing them, you need to watch, and hope for, progress with VHDL. Even the biggest IC house won't be able to deal with a 3000-page schematic. If they can't put the functions on their chip schematics, you're going to have to put them on yours.+

SOURCE

Synopsys, Inc.
1098 Alta Avenue
Mountain View, CA 94043
(415) 962-5000

Synopsys offers VHDL tools, documentation, and training.

Tom Cantrell holds a B.A. in economics and a M.B.A from UCLA. He owns and operates Microfuture Inc., and has been in Silicon Valley for ten years involved in chip, board, and system design and marketing.

IRS

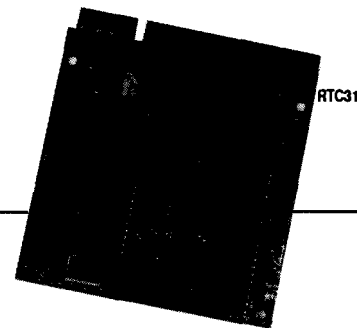
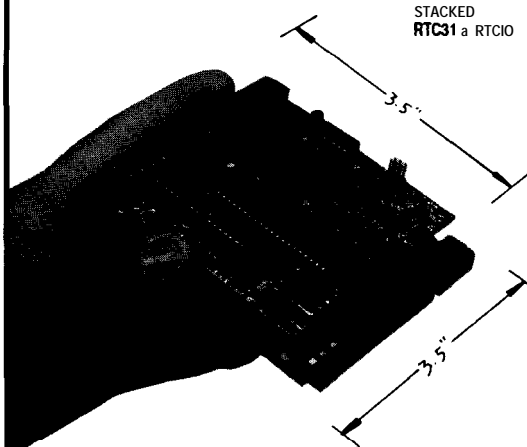
274 Very Useful
275 Moderately Useful
276 Not Useful

MICROMINT Introduces "Micro" Controlling!

After years of experience in manufacturing OEM controller boards and talking to customers, we think we have hit upon just the right combination of format and function to satisfy even the toughest case of "relay mentality." Realizing that not every computer/controller application warrants a Cray XMP, Micromint offers a tiny 8031/8052-based controller board for those dedicated and cost-sensitive installations.

New **MC-NET™** software links your desktop up to 31 RTC controllers.

STACKED
RTC31 a RTCIO



RTC31 and RTC52 Technical Specifications

- 9031 processor (RTC31) or Micromint 80C52-BASIC processor (RTC52)
- 11.05-MHz system clock
- Uses 8K or 32K memory chips
- Up to 64K bytes of RAM or EPROM
- 5-volt-only operation
- 11 0-19200 bps RS-232 and/or RS-495 serial port
- Use stand-alone or networked
- 12 bits of parallel I/O
- Vertical-stacking expansion bus
- Screw terminals or quick disconnects
- Small 3.5"x3.5" format
- 30 mA typical operating current (RTC52)

RTC31-1 8031 Controller	\$119.00
RTC31-1 OEM 100-Quantity Price	\$79.00
RTC52-1 80C52 Controller	\$139.00
RTC52-1 OEM 100-Quantity Price	\$99.00

RTCIO Technical Specifications

- Three bidirectional parallel ports (24 bits)
- 8-channel, 8-bit ADC (0-W); 9,000 samples/sec (optional B-channel, 10-bit ADC)
- 4-channel, 8-bit DAC (0-5V); 2-µs response time
- Battery-backed dock/calendar and presettable time-interrupted capability
- DC to DC conversion—5-volt-only operation
- Screw terminals or quick disconnects
- Small 3.5"x3.5" format

RTCIO RTCIO board with parallel I/O and ND converter	\$129.00
RTCIO OEM 100-Quantity Price	\$89.99

Also Available

RTC-OPTO a-channel Optoisolated I/O Expansion Board	single qty. \$139.00
RTC-SIR Serial, Timer, and Infrared I/O Expansion Board	single qty. \$149.00
ATC-LCD LCD, Keyboard, and X-10 I/O Expansion Board	single qty. \$99.00
RTC180-1HD64180 Controller Board. 96K memory; 1024 bits EEPROM; 24 bits TTL I/O; a-channel, 10-bit ADC; 2 serial ports	starting at \$239.00



MICROMINT, INC.

4 Park Street, Vernon, CT 06066
Tel: (203) 871-6170 • Fax: (203) 872-2204

PRACTICAL ALGORITHMS

Scoff Robert Ladd

Around and Around We Go...

During a recent solo backpacking trip into the mountains around my home, I began to think about how my life has involved circles. I began my college career in physics, moved into astronomy, and gradually became enamored of computers. As I've worked with computers, the applications I've written have gradually brought me full circle back to my roots in science. Computers make excellent tools for studying the universe; programs can simulate a nuclear reaction, the collisions of galaxies, or the formation of a hurricane-without the damage incurred by doing these things for real.

If you've had any contact with computers in recent years, you'll have encountered the term "fractal." A fractal is a geometric object which is defined by an iterative or recursive algorithm. Traditional (so-called Euclidean) geometric forms like ellipses and rectangles are defined by simple formulas. Euclidean objects are useful in describing human-made objects, but they are totally inadequate for rendering natural things. Fractals can be used to precisely depict natural objects like the clouds, trees, and mountains. Like Ezekiel's space ship, the world consists of wheels within wheels. And so do our programs.

In the last column, I began a presentation on recursive algorithms. If you've spent much time studying programming algorithms, you'll note that many of them involve recursion. In many cases, recursion is the simplest and most direct method of implementing an algorithm. However, recursion isn't the only way in which many of these algorithms can be implemented; recursion can always be replaced by iteration. In almost all cases, the recursive implementation of an algorithm will be slower and smaller (in terms of lines of source code) than the iterative version.

The factorial in the last column was an example of this principle. A factorial calculation is pretty simple, and it's obvious how it can be implemented either recursively or iteratively. What about more complicated algorithms?

Listing 1 shows a classic recursive implementation of the famous Quicksort algorithm for an integer array. Quicksort has become the preferred sorting algorithm due to its relative simplicity and very fast run-time. It operates on the divide-and-conquer theory: it breaks up the problem of sorting into pieces which can be handled simply. The internal `QSWORK` function does all the work; it works a contiguous section of the array called a partition.

```
PROCEDURE QuickSort(VAR a : ARRAY OF CARDINAL);
PROCEDURE QSWork(l, r : CARDINAL);
VAR
  x, temp, i, j : CARDINAL;
BEGIN
  i := l;
  j := r;
  x := a[(l + r) DIV 2];
  REPEAT
    WHILE a[i] < x DO INC(i) END;
    WHILE a[j] > x DO DEC(j) END;
    IF i < j THEN
      temp := a[i];
      a[i] := a[j];
      a[j] := temp;
      INC(i);
      DEC(j);
    END
  UNTIL i >= j;
  IF (j+1) < r THEN
    QSWork((j+1), r)
  END;
  IF l < j THEN
    QSWork(l, j)
  END
END QSWork;
BEGIN
  QSWork(0, HIGH(a))
END QuickSort;
```

listing 1 -The recursive Quicksort algorithm operates on the divide-and-conquer theory.

The ends of the partition currently being sorted are represented by `QSWork l` and `r` parameters. Quicksort starts the process by calling `QSWork` with `l` pointing to the first element of the array and with `r` pointing to the last element in the array. The variables `i` and `j` represent the currently selected items in the left and right sections of the current partition. Variable `i` begins on the left end of the partition, and `j` begins at the end of the partition. A comparator, `x`, is selected; it can be any of the numbers in the current partition, and in this case I use the value stored in the middle element of the partition. The selection of the comparator is very important, as I'll explain in a moment.

In the inner loop of `QSWork`, `i` moves to the right and `j` moves to the left, looking for values which are not on the correct side of the partition based on the value of the comparator. The comparator `x` is compared with the elements pointed to by `i` and `j`. Variables `i` and `j` move toward each other, swapping partition elements when necessary. When `j` is less than `i`, all elements less than the

comparator are stored in the partition elements 1 through j , and all of the elements higher than the comparator are stored in the elements i through r . At this point, the partitions 1 through j and i through r are sorted using recursive calls to `QSort`. Eventually, the partitions are only one element in length, and the array has been sorted.

Quicksort is perhaps one of the most elegant algorithms in computing. It has only one drawback: The selection of the comparator for a partition has a strong effect on the time Quicksort takes to do its work. The ideal comparator is the median value of the elements in the partition. If the comparator is too small or large, it will cause Quicksort to perform slowly. In the next column, I'll discuss how you can determine the best comparator value for a given set of input data. For now, selecting the item in the middle of the partition works very well in almost all situations.

A recursive implementation of Quicksort is found in nearly every text on algorithms because it's simple and obvious. Simple and obvious, yes-efficient, no. Function calls have overhead, and a Quicksort on a large array will do many, many function calls. In addition, some programming languages do not support recursion. Removing recursion from Quicksort is relatively easy, yet very few resources explain exactly how this is done.

Listing 2 shows an iterative implementation of Quicksort. It's virtually identical to the recursive implementation, and simply replaces the recursive calls to `QSort` with a pair of loops. The inner loop processes left-side partitions, represented by the range 1 through r . Variable r is reset at the end of this loop to j , performing the same action as the recursive call for `QSort(1, j)`. The values of $j+1$ and r are stored in the stack array, which is used by the outer loop to set up partitions on the right side of the array. When the stack is empty (i.e., s equals 0), no more partitions remain to be reorganized, and the sort is done.

In the case of Quicksort, the iterative version improves sorting performance by about 40%, in spite of the fact that the iterative version is longer and seemingly more complex than its recursive relative. The savings comes from eliminating the need to push arguments and perform function calls. That's a pretty important difference when sorting is a primary task in your programs. In my standard libraries, I implement only the iterative **version** of Quicksort.

IN THE MAILBOX...

A few readers pointed out that the factorial of a number can be calculated very quickly through the use of a look-up table. The table is basically a list of the factorials for numbers in a predetermined range. This is very fast, since no calculations need to be performed. However, the table uses up memory, and can only contain a certain set of values. Of course, there are ways of making the table more memory-efficient and flexible. Look-up tables are a useful programming tool, which I'll be covering in the future.

There have been a few requests for a presentation of math algorithms in this column. How does a computer calculate the sine of an angle or determine a square root

```

PROCEDURE QuickSort(VAR a: ARRAY OF CARDINAL);
CONST
  stackSize = 64;
VAR
  stack : ARRAY [1..stackSize] OF
    RECORD
      l, r: CARDINAL;
    END;
  i, j, l, r, x, temp, s: CARDINAL;
BEGIN
  s := 1;
  stack[1].l := 0;
  stack[1].r := HIGH(a);
  REPEAT
    l := stack[s].l;
    r := stack[s].r;
    DEC(s);
    REPEAT
      i := l;
      j := r;
      x := a[(1 + r) DIV 2];
      REPEAT
        WHILE a[i] < x DO INC(i) END;
        WHILE a[j] > x DO DEC(j) END;
        IF i < j THEN
          temp := a[i];
          a[i] := a[j];
          a[j] := temp;
          INC(i);
          DEC(j);
        END
      UNTIL i >= j;
      IF (j+1) < r THEN
        INC(s);
        stack[s].l := j + 1;
        stack[s].r := r;
      END;
      r := i;
    UNTIL l >= r
  UNTIL s = 0
END QuickSort;

```

Listing 2—The iterative version of Quicksort replaces the recursive calls with a pair of loops

without using a math coprocessor? That'll be the subject of some future columns as well. Another reader was interested in knowing more about heaps and priority queues. Someone else suggested that I cover advanced linked-list techniques. I'll get to all these topics as time goes on.

Some of you would like to see me to replace Modula-2 with another language, such as C or Pascal. I've talked with `CIRCUIT CELLAR` INK's editors, and we all seem to agree for now: Modula-2 stays. This column is about algorithms, not a specific programming language. Presumably, you'll be using the ideas and concepts presented here, as opposed to directly copying the program code verbatim. Modula-2 is more standard than Pascal, and is far clearer than C. I will keep noting your suggestions, though.

As always, I'm interested in your views. If there's something you'd like to see me cover, feel free to drop me a line. When an explanation I've given isn't clear enough, complain! If I'm not providing useful information, then I'm not doing my job. So, until next time. ...

Scott Ladd is a writer specializing in computer software. Correspondence concerning "Practical Algorithms" may be sent to him at: Scott Robert Ladd, 705 W. Virginia, Gunnison, CO 81230, (303) 641-6438.

IRS

277 Very Useful
278 Moderately Useful
279 Not Useful

Going for the Gold...

CIRCUIT CELLAR INK's Second Design Confesf Winners Shine with Design and Execution Quality

The entries are in, the judging is over, and it's time to celebrate the winners of the Second CIRCUIT CELLAR INK Design Contest. Our judges sifted through several thousand pages of design documentation, schematics, and code listings to arrive at the winners of this year's contest. The judging was made more difficult by the generally high level of entry quality this year. Our thanks and congratulations go to all contestants, whether or not they're listed by name on these pages.

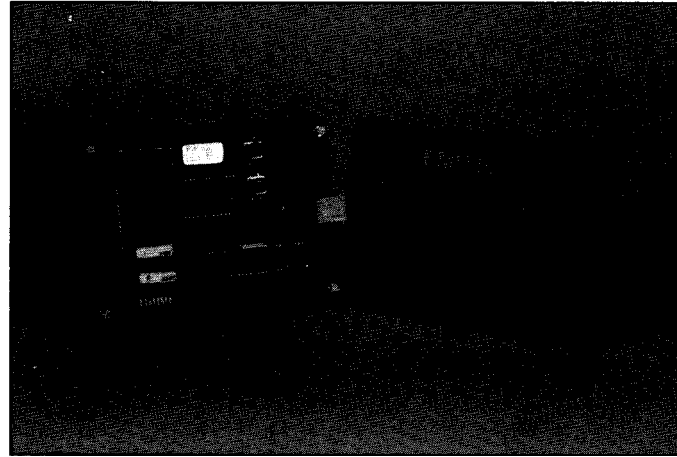
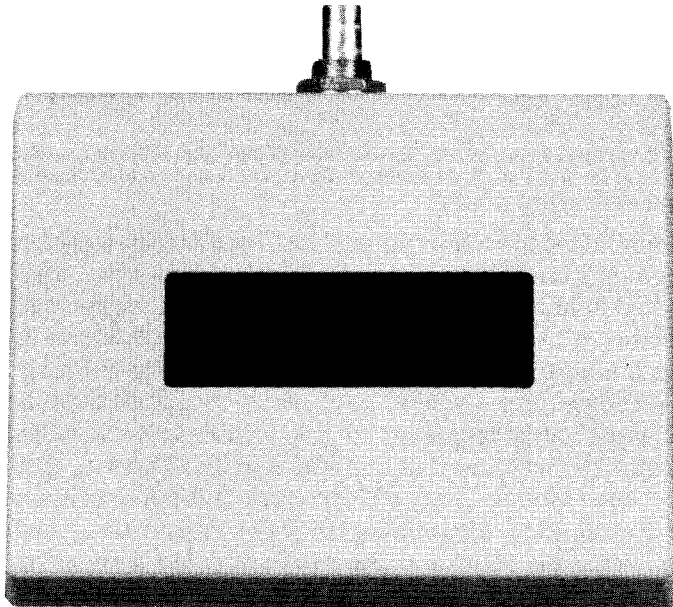
Several winners have already agreed to write extended articles on their projects, so we're looking forward to a year's worth of high-quality additions to the CIRCUIT CELLAR INK project roster. Stay tuned, and start thinking about your entry for the Third CIRCUIT CELLAR INK Design Contest!

First Place, Cost-Effective Category: \$500

A Time-Domain Reflectometer

by John Wet-troth and Brian Kenner

John and Brian took first place for their practical 8051-based solution to checking network cabling for integrity and termination using signal reflection characteristics. Their elegant touch-screen package won the hearts (and points-total) of our judges.



First Place, Open Category: \$500

EMU3x, an 8031 emulator

by David Wickliff

EMU3x took top honors in the Open Category by giving users an 8031/8032 In-Circuit Emulator at a reasonable price and level of complexity. EMU3x is the basis for a top-drawer development system, and a fitting winner of the contest.

Honorable Mention, Open Category: \$50 + 1-year subscription

A VCR Data Backup Card

by Winifred Washington

Data backup is an important part of any working computer discipline. With the VCR Data Backup Card, users can take advantage of a reliable tape drive that many people already have in their homes. A PC-bus computer backs up to a standard home VCR using this board with its UPI-41A (enhanced 8748) controller.

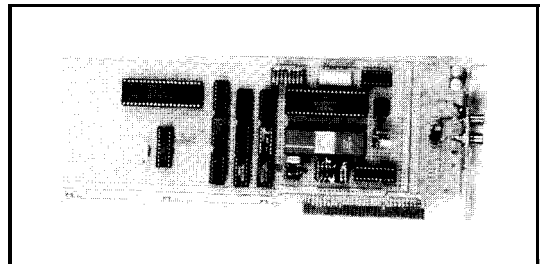
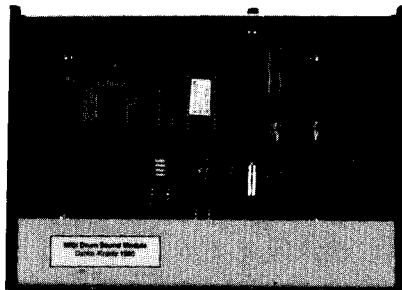
Honorable Mention, Cost-Effective Category: \$50

+ 1-year subscription

MIDI Drum Sound Unit

by Tom Dahlin and Don Krantz

This MIDI-controlled drum unit uses digitally sampled sounds to provide full drum kit performance. Don and Tom use the National Semiconductor HPC46003 to control a high-quality drum unit for MIDI-music fans.

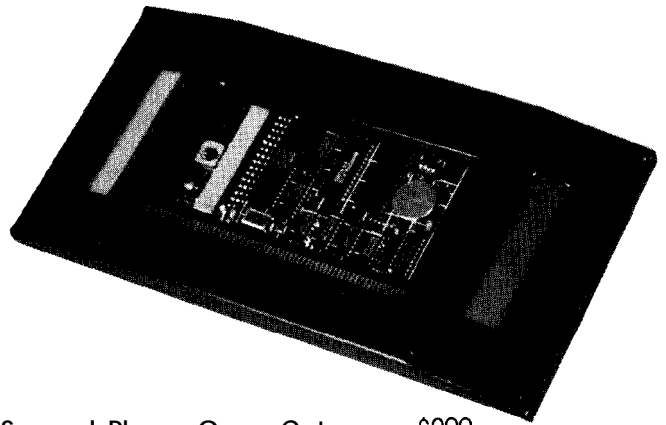
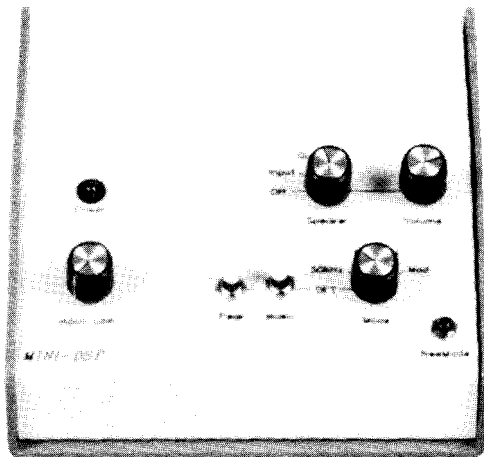


Second Place, Cost-Effective Category: \$200

Mini-DSP

by Steven Reyer

Digital signal processing is one of the 'hottest' topics in digital electronics today, and this experimenter's box, based on the TMS320E 15 DSP with on-board EPROM from Texas Instruments, allows low-cost experimentation with DSP techniques.



Second Place, Open Category: \$200

Crib-puter

by Craig Anderson

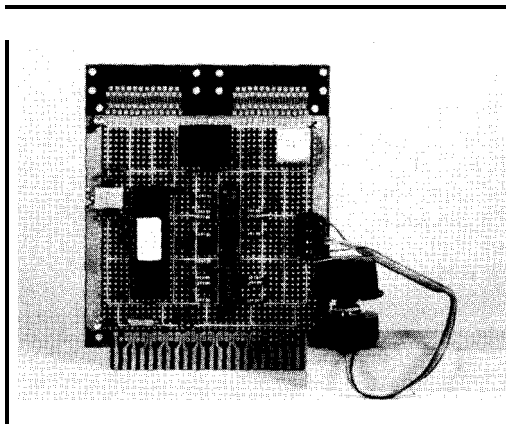
The Crib-puter Cribbage Computer impressed the judges with its design, and delighted them with its execution. Craig used the 80C52-BASIC controller as the heart of a beautifully built portable cribbage companion. LCD displays and automatic light-level sensing completed a great design package.

Third Place, Cost-Effective Category: \$100

An Electronic Combination Lock

by David Penrose

An electronic version of the old-fashioned combination lock scored high points for elegance and functional simplicity. An 8748 provides the brains for this project while an optical encoder provides the clever 'combination dial.'

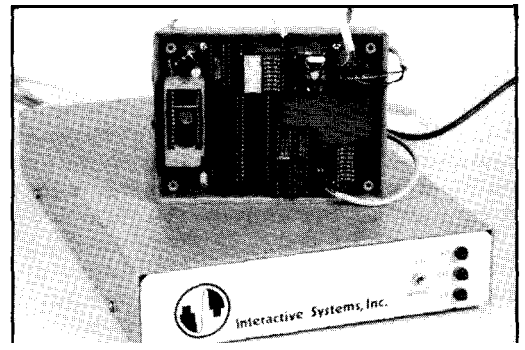


Third Place, Open Category: \$ 100

Video Editing Controller Modules

by William Kressbach

A Zilog Z8 provides the computing horsepower for this system of video editing controllers. The complete system allows insert or assemble editing, calculates durations, records time code information, displays status information, adds titles, and provides time code locking. In all, a most impressive set of professional-quality video editing tools.

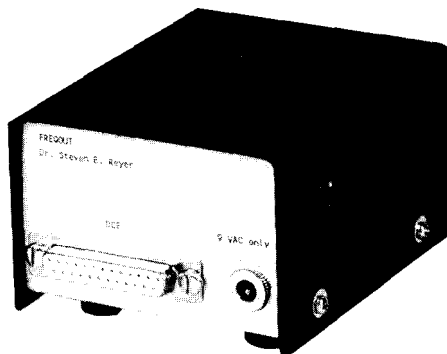


Honorable Mention, Cost-Effective Category: \$50

+ 1 -year subscription

FREQOUT, a power-line frequency monitor

by Steven Reyer



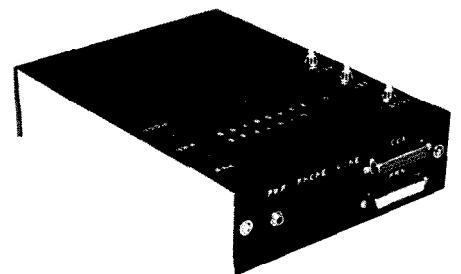
Steven's second entry captured Honorable Mention by providing a useful function in a practical package. This design uses an Intel 8748 to check 50- or 60-Hz AC power lines for frequency on a continuing basis.

Honorable Mention, Open Category: \$50 + 1-year subscription

TeleLogger

by Ed Daly

If you have teenagers with long-distance friends or a business where telephone calls are billable expenses, the TeleLogger could make your life much easier. This 87C51-based device logs the destination telephone number and duration of all telephone calls. The logged data can be either downloaded to a host computer via RS-232 lines or printed on a parallel printer. Useful, well-designed, and well-built—the TeleLogger typifies the CIRCUIT CELLAR INK Design Contest entries for 1990!



CONNEC- TIME

Conducted by
Ken Davidson

Excerpts from the Circuit Cellar BBS

The Circuit Cellar BBS
300/1200/2400 bps
24 hours/7 days a week
(203) 871-1988
Four Incoming Lines
Vernon, Connecticut

The data logger project which Steve presented in the Building Automation special section in the June/July 1990 (#15) issue of CIRCUIT CELLAR INK generated quite a bit of traffic on the Circuit Cellar BBS. The first discussion contains excerpts from several of those message threads. Also covered in this installment of "ConnectTime" is the use of X-20 devices in a low-performance LAN, and how to decide if a processor has enough horsepower to handle the task assigned to it.

Msg#:29497

From: BOB PADDOCK To: STEVE CIARCIA

Your comment in your Data Logger article reminded me of one those things "that drives you positively nuts."

Something I learned the hard expensive way is that when troubleshooting a circuit that has the RS-232 connector hooked up, make sure you know how the power supply of the computer driving the RS-232 port is wired!

I was working on a PC board that was supposedly isolated from the AC line by its power transformer. I touched my scope probe to the board and ****BANG****! After saying a few words of thanks about safety glasses and picking the shrapnel out of my hair, I sat there wondering how that happened. I came to find out that the Epson Equity computer that I was using has its RS-232 pin-7 logic-ground tied to its AC neutral line. I had a path back through the building wiring to my supposedly isolated circuit.

Does anyone know if this is a common thing to do (tie logic ground to AC neutral) in computers, or is this just A Dumb Thing that Epson did?

I think you missed covering (or did I miss you covering it?) something important in Figure 4 for the fixed-interval trigger. That is that the two unused sections of the 4070 are not shown as being tied off to GND or +3V. The schematic leads you to believe they are not connected at all. I'm sure that we both know that it is an absolute must to NEVER leave a floating input on a CMOS gate when trying to get the absolute minimum power consumption, but some of the magazine readers might not realize this.

Also something else that I noticed (something that my drafting teacher kept pounding into my head) is that the longest plate of

the battery is always the positive end (the 8.4V NiCad pack and the solar cells of the light intensity sensor are shown with reversed polarity), and that the curved end of a capacitor, whether polarized or not, goes to the most negative potential (ground, in this case). The 33-pF caps on the MM5369AA are shown backwards

Was my drafting teacher wrong?

Msg#:29516

From: DALE NASSAR To: BOB PADDOCK

Did you say the RS-232 ground was connected to AC neutral? This could be a very hazardous situation considering that many households have neutral and hot reversed!! :-(

Msg#:29527

From: BOB PADDOCK To: DALE NASSAR

Yep, that's what I said. I even went and checked several other Epson computers around the plant here, they were all the same way. All we have is Epsoms so I don't have anything else to compare to.

Msg#:29586

From: TIMOTHY TAYLOR To: BOB PADDOCK

This is off the top of my head, but I *think* the spec for RS-232C says that pin 7 is signal ground and pin 1 is tied to earth ground. Some folks (maybe most) leave out pin 1 entirely. Also, there are a lot of folks out there that I've seen connect 1 and 7 together. Most equipment seems to connect 7 to signal ground which a lot of times ultimately gets connected to chassis ground, thus also earth ground. I don't see a big problem in this. I have seen many houses (new and old) that have hot and neutral reversed. Ground, however, is a rarity to have reversed with the hot lead. If it is, you'll have discovered the problem well before you plug in your communications gear.

Years ago I worked for a company that routinely tied earth ground and neutral together at the supply. When the system was plugged into an improperly wired outlet, many, many **bad** things happened. I still think that system lies in a corner somewhere.

So, after much rambling, I don't think it's a wise thing to connect neutral to chassis (or signal) ground. The outlet's gotta be perfect or look out!

Msg#:29508

From: KEN DAVIDSON To: BOB PADDOCK

Oops. You're right about the battery. I probably should have picked up on that. The long end is always the positive terminal. As for the curved end of the caps, that's something we've never worried about much when representing nonpolarized caps. Aesthetically speaking, you are correct, but how the circuit is ultimately wired doesn't depend on it.

Msg#:29790

From: STEVE CIARCIA To: BOB PADDOCK

The unused pins were indeed tied to ground. I just forgot to mention it. Regarding long and short lines on batteries, there are people (believe it or not) that think the convention is the other way and that the long line designates the large "Earth" sink when that pole is ground reference. Since I can't seem to agree with either I physically put a "+" sign so I don't screw it up. And, since I tend to use two straight lines for caps, polarized or not, direction is moot.

Msg#:29894

From: ED NISLEY To: KEN DAVIDSON

I think the capacitor **symbol** dates back to the old **Leyden jar days**; the curved symbol was the foil wrapped around the outside of the jar.. . which ought to be close to ground if you have any sense at all!

On the other hand, the foil on the inside of the jar was curved, too, so what 'cha gonna do?

Msg#:29099

From: LEE AH0 To: STEVE CIARCIA

I really liked your article on the data logger. I have several uses for one, so I'm going to build my own. In the article, you mentioned monitoring the **pH** level of the water in a brook. Is there some type of sensor that I can get to make this measurement? Or, can I make one? How about sensors to measure humidity?

Msg#:29205

From: STEVE CIARCIA To: LEE AH0

All the **pH** sensors I've seen are pretty expensive but the humidity one is fairly cheap. Ken Davidson might still have the part number for the one we sampled. Also, many moons ago we had quite a bit of conversation here on sensors. Perhaps you should peruse the archives.

Msg#:29214

From: LEE AH0 To: KEN DAVIDSON

I asked Steve about **pH** and humidity sensors for use with the data logger he described. He said you might still have the part number of the humidity sensor that you experimented with. Can you tell me where I could possibly get one?

Steve also said that all the **pH** sensors that he knew of were pretty expensive. Is it possible to construct one?

Msg#:29241

From: KEN DAVIDSON To: LEE AH0

Actually, the humidity sensor we have (which, by the way, I haven't tried yet) was obtained through a group purchase that another BBS user set up around September '89. It was long enough ago that I doubt any messages about it are still active and on-line.

It is a Philips 2322-691-90001 relative humidity sensor that was featured in the February 1986 issue of "Radio-Electronics." It is actually a variable capacitor which has a value determined by the relative humidity. You need a circuit which converts the capacitance to something a computer can use.

I don't have any information at all on **pH** sensors.

Msg#:30357

From: PELLERVO KASKINEN To: LEE AH0

Well, you might be able to make a **pH** sensor yourself, provided you know how to make porous glass. What you need for the electrodes are Calomel and the porous glass (or ceramic) electrode that contains a saturated solution of **KCl**. The porosity must be such that ions flow through, but you do not let the solution as such disappear too soon. Typically, in dirty environment a higher loss of the electrolyte is required and tolerated. The slow migration of ions through this porous plug is also the reason for a very high resistance that the electrodes exhibit, requiring even higher resistance for the measuring amplifier-in the thousands of megohms! And of course, you have to correct the **measurement** for the temperature effects (i.e., you should include a temperature sensor in the electrode construction).

Did I whet your appetite?

Msg#:29354

From: FRANK KUECHMANN To: LEE AH0

One good **pH** sensor is a **pH** electrode; **they're** generally expensive, delicate and available through scientific supply houses. Use with a J-FET input op-amp, read the amp's output with an ADC or voltmeter. A source for the electrode and example circuits is Vernier Software (2920 SW 89th Ave., Portland, OR 97225; 503/297-5317). Cost is rather high from Vernier (~\$30) and you can probably cut it in half if you track the thing down elsewhere.

The group purchase of the Philips humidity-sensitive capacitor Ken mentioned was from Newark Electronics at a cost of about

\$7 per capacitor. The reason a group purchase was made is Newark's \$50 minimum order.

The Philips data sheets available from Philips (and perhaps Newark) show a lot of stand-alone measurement circuits and other applications; several can be easily adapted for use with computers using either an A/D converter or a frequency (pulse) counter.

Msg#:29604

From: LEE AH0 To: FRANK KUECHMANN

I am building the data logger, but I'm using the MC68HC11 micro from Motorola. I know that might be a sin amongst most of the other BBS users, considering the strong Intel 8051 family following, but for applications like this I prefer it.

I was hoping I could get some info on that sensor before I ordered some. Thank you.

Msg#:29623

From: FRANK KUECHMANN To: LEE AH0

I like the 68HC11 and Moto's other processors like the 6809 and 68000 series, but for a lot of the data-logging applications I'm involved with something like the 80C52-BASIC interpreter is better suited than anything I know of with a Moto processor. I've heard there are a decent FORTH interpreter or two for the 68HC11, but haven't had a chance to try 'em.

X-10 control is always a popular topic, and with two-way devices like the TW523 starting to show up, it's tempting to look at using the power line for a low-performance LAN using X-20 devices as transceivers. Closer inspection reveals that it may not be such a good idea.

Msg#:30273

From: MARK BALCH To: KEN DAVIDSON

Hi Ken. I just looked back at issue #5 of Circuit Cellar INK for your article on the X-10 TW523 power line transceiver and it's gotten me interested. I have an idea for an X-10 network. I'd like to do some small-scale stuff and I don't want to run RS-485 wires through three floors of house, so X-10 sounds perfect, but I need to know some facts first.

What I plan to do is to have a master controller configured as a hub that continually polls other controller connected through X-10. This way, any slave can talk to any other slave through the hub. Then I figured that I could run small, local RS-485 networks off of each slave controller. I'd have one for the attic, one for my room, and so on. This will prevent excessive X-10 loading as far as nodes are concerned.

First of all, does this sound realistic or am I overlooking a key obstacle that will fry my house and anything in it? I wrote to X-10 (USA) today for the same info, but I'd like to get your opinion since then I may build the transceivers on my own from the schematics that you had in your article. Thanks.

Msg#:30293

From: KEN DAVIDSON To: MARK BALCH

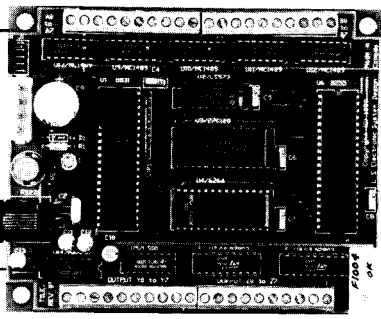
Two limiting factors here are going to be maximum data rate and error (noise) recovery. The second kind of depends on the first. With valid X-10, you send bits only on AC line zero crossings, so at the outset your maximum throughput is only 120 bps. Then you send each bit twice: once in its normal state, then again in its complemented state on the next zero crossing. That cuts you down to 60 bps, but you gain a bit of error checking (that's checking-not correction). You could probably leave it at that if you were only talking between a pair of computers with no module control, but if you want to stay with a valid X-10 format, you have to begin each packet with a 4-bit start code, then you can send out nine data bits ($4 + 9 * 2 = 22$ bits all together). Then there has to be at least three zero crossings before you can send the next packet.

The TW523 checks incoming packets for errors and only passes along valid packets. Therefore, you have to incorporate a fairly robust protocol in whatever you're doing to allow for lost packets, timeouts, and so on.

You could probably make a very (*very*) low performance network with X-10, but I'm not sure you'll find the performance acceptable.

As for building your own modules, I really wouldn't recommend it. We're not talking 5 VDC here. At less than \$20 (I think) each assembled, tested, and UL approved, you really can't beat it. The schematic was published for informational purposes only.

DC31
8031 BASED CONTROLLER BOARD



- 8031 microcontroller running at 11.0592 MHz
- 16K EPROM and 8K SRAM JEDEC sockets
- 8255 parallel I/O, MAX232 for RS232C serial I/O port
- ULN2803A output drivers - 16 lines ± 30 V input buffers - 20 lines. Screw terminals on 32 I/O lines
- Full schematic diagram, operating manual on disk

KG31 controller board assembled without EPROM \$129.00

ControllerKITDG31K: PC board and all components except screw terminals, RAM and EPROM \$70.00

:PROM27C128 CMOS programmed with TILE firmware (Programmable controller with Real Time Clock) \$20.00

:PROM27C64 CMOS programmed with DG31M and disk (IBM format) with monitor program, modules source code listings and manual \$30.00

IOME AUTOMATION and SIMPLIFIED SECURITY SYSTEM — complete project using TILE controller and low cost electronic modules (shipping included) \$10.00

L.S. ELECTRONIC SYSTEMS DESIGN
280 Camilla Rd. • Mississauga Ont. L5A 2J8 Canada

TERMS: Shipping US/Canada \$6. Check or money order please.

Msg#:30565

From: MARK BALCH To: KEN DAVIDSON

Gosh, that **kinda** puts a damper on my plans! Thanks for the info. I suspected that the bit rate would be slow because of the 60-Hz line frequency, but I didn't have your first article in the series. With <60 bps I'm not going to start with any X-10 plans because I have no use for dimming lights at the moment. One of these days, I'll string some twisted pair cable and go with RS-485.

The first task in designing a system with a microprocessor attached is to decide whether the processor chosen can handle the task. What's the best way to go about making that decision without building the system and trying?

Msg#:27296

From: JOHN OLIVA To: ALL USERS

What are some techniques which have successfully been used to determine whether some particular real-time processor (general purpose or DSP) can actually handle the specified task in real time? If the algorithm/task has previously been implemented on a processor it is possible to analyze the number of machine cycles on that processor so **that** an estimate of the corresponding number of cycles on another processor can be made. Doing this requires an analysis of the types of instructions allowed by each processor and the execution speed of these instructions. My question is aimed more at the case where the algorithm/task has never been implemented but some system performance levels are known a priori.

I know that the answer to this question is very situation specific but some techniques and/or guidelines would be helpful. Thanks in advance.

Msg#:27483

From: ED NISLEY To: JOHN OLIVA

Well, I'll wave my hands for a while if you folks promise not to laugh. This may sound a little oversimplified, but it works reasonably well in practice. OK?

It takes 100 machine instructions to do any useful, nontrivial operation. If you have one such operation, your effective throughput is 1% of the MIPS rating (and it's a good idea to weight the MIPS to average instruction duration rather than just the register-to-register ops!). If you've got several such operations that must be **done** in a specific amount of time, add 'em up, multiply by 100, divide by the specified time to get MIPS.

If you know enough **about** the problem to separate it into "interrupt code" and "normal code" you can figure the interrupt code using that rule of thumb and then decide how much is left over for the rest of the code. If there isn't enough left over, you're in trouble.

For higher-level languages, you need to take a look at the compiler's output and decide what derating factor to apply. Based on some 8031 tinkering I've been doing recently, it looks like the right number is about 10. **so** your "useful, nontrivial operation" had best be expressible in about 10 lines of C code! You can work the other way and decide that 100 lines of C will produce about **1000** machine instructions, then get the performance data from there.

If I size up a project like this (being maybe a **leettle_ _bit_** more formal about it), measure it against the performance requirements and the "desired" processor, and find that we're within a factor of two or three from running at 100% capacity, I get real worried.

Now, I can hear the outraged screams from here. OK, party people, what rules of thumb do **_you_** use?

Msg#:27502

From: NATHAN ENGLE To: ED NISLEY

I promise (ha ha) not to laugh (**hohoho**) but that's just about the advice that I would go with; it's all very well for the salesmen to tell you how easy their favorite processor is for RT development, and how easy it is to shoehorn everything in and still be very efficient. Hogwash.

Ed is right that if you think you're within a factor of two or three you're probably **doomed**. Unless your application is VERY simple and has nothing else going on but one very high priority task, it's likely that you'll lose that two-to-three times margin for error very fast.

The sorts of things I've been doing recently haven't been what I would call stretching the limits of my CPU; my phone line simulator is really only required to time pulses to tens of milliseconds. However, I have been looking at ways of generating precise waveforms for my 350 Hz, 440 Hz, and so on, and I was considering doing them in software. Doing one tone seemed pretty easy since I had a free timer, but doing all four of the ones that I need turned out to **be** out of the question on my system; the overhead of jumping in and out of interrupt routines as the edges changed was going to bury all my other processes. The thing I've settled on now is just a bunch of LS393 counters that divide down a crystal (i.e., a complete hardware solution that doesn't attempt to do anything in software).

Moral of this story: if you have a half dozen "easy" things to do, then it's not necessarily easy to do them all at the same time.

I like Ed's factor of 10 for safety; I know that people around here are concerned with being very efficient and economical, but there are some places where economy should be set aside until you figure out how much CPU punch you really need.

Msg#:27575

From: JOHN OLIVA To: ED NISLEY

Thanks for the reply. I have read your column in Circuit Cellar INK and have read your advice on this BBS for a while now and very much respect your advice. I found your comment about the

compiler's efficiency to be of great interest. I recently took a short course on firmware programming where the instructor indicated that a good compiler coupled with a good firmware programmer could yield code that could be derated by a factor of three over the same code done in assembly language. I have discussed this with several of the people I work with and they find that hard to believe. My own experience using a C compiler to generate code in comparison with doing the task in assembly language favors your factor of 10 estimate.

I have also noticed that **many** of the optimization techniques that are used by the available compilers don't necessarily produce faster code, but will usually produce shorter code. When you take on a real-time coding project, do you first attempt to write it in a high-level language or do you head write for the low-level bit banging?

Msg#:27643

From: ED NISLEY To: JOHN OLIVA

There are (at least!) two types of "real-time" systems: little bitty ones where you're concerned with precise timings, and big ugly ones that have lots of tasks, real operating systems, oodles of software, and scads of programmers. I've got more experience with the former, so anybody who thinks I'm shortchanging the latter is quite correct.

For anything that requires precise timing (measured in microseconds or very accurate milliseconds), there isn't any choice in the matter-you start right out with assembler.

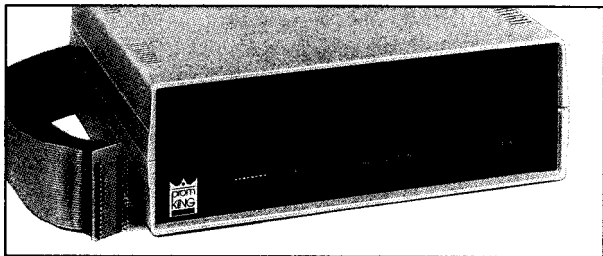
Although the conventional advice is to write it in a "real" HLL and then recode the hot spots in assembler, I find that it's tougher to pull out a hunk of code and make it faster after it's been designed to work another way. If it's a real critical piece of code, you just can't tack on all the subroutine call overhead in the most logical place...you won't get there in time!

For example, we're working on a high-speed **modem** sort of thing that's controlled with (what else?) an 8031. The user interface part of the code will be in C, but the bit fiddling must be in assembler because we've only got 75 instructions or so between incoming bits.. .

The 8031 isn't a really **good** machine for high-level languages. It's got a bunch of internal RAM that holds the processor stack, but nearly all "real" variables must be in external RAM. Hey, that sounds like a RISC machine, doesn't it? Hmm.. .

The Circuit Cellar BBS runs on a 10-MHz Micromint OEM-286 IBM PC/AT-compatible computer using the multiline version of The Bread Board System (TBBS 2.1M) and currently has four modems connected. We invite you to call and exchange ideas with other Circuit Cellar readers. It is available 24 hours a day and can be reached at (203) 871-1988. Set your modem for 8 data bits, 1 stop bit, and either 300, 1200, or 2400 bps.

STOMP OUT EPROM MADNESS



The PROM KING emulates EPROMS, saving both time and money during your development cycle. Programmable in seconds via your PC printer port or any computer RS232 port, it can emulate most 27xxx devices.

- 8K-8M bit devices
- High speed download:
 - Universal RS232
 - PC printer port
- Menu driven software
- Battery backup
- 8-256 bit downloads
- Easily expandable:
 - 4 EPROMS per unit
 - Up to 8 units
- Also programs like a real EPROM

\$599 for 150nS units with 256K bits Ask for pricing of other options,

Made in USA by

TRAXEL LABS INC.
 BOX 239 • RONKONKOMA, NY • 11779
 516•737•5147 FAX•516•737•0349

IRS

- 280 Very Useful
- 28 1 Moderately Useful
- 282 Not Useful

SOFTWARE and BBS AVAILABLE on DISK

Software on Disk

Software for the articles in this issue of Circuit Cellar INK may be downloaded free of charge from the Circuit Cellar BBS. For those unable to download files, they are also available on one **360K**, 5.25" IBM PC-format disk for only \$12.

Circuit Cellar BBS on Disk

Every month, hundreds of information-filled messages are posted on the Circuit Cellar BBS by people from all walks of life. For those who can't log on as often as they'd like, the text of the public message areas is available on disk in two-month installments. Each installment comes on three **360K**, 5.25" IBM PC-format disks and costs just \$15. The installment for this issue of INK (October/November 1990) includes all public messages posted during July and August, 1990.

To order either Software on Disk or Circuit Cellar BBS on Disk, send check or money order to:

Circuit Cellar INK — Software (or BBS) on Disk
 P.O. Box 772, Vernon, CT 06066

or use your **MasterCard** or Visa and call (203) 875-2199. Be sure to specify the issue number of each disk you order.

STEVE'S OWN INK

Steve *Ciarcia*

A Computer is A Computer

The power of magazine advertising is wonderful. It's America's way of making all companies look equal in the eyes of the public. Where else can you, as a start-up company, have an equal voice to Lotus or Microsoft? Where else can the potential response to the right product be like hitting the lottery? How many of you realize that a major software manufacturer started the whole place with a ninth-page ad for a \$49 Pascal?

Sorry if I sound like I'm pushing the idea; I'm really just sitting here chuckling to myself. For the first time, I'm actually an advertiser too. Let me explain.

As many of you probably know, I presented hardware design projects in *BYTE* for a dozen years or so. During that time a substantial engineering staff evolved to support these projects and it is still with me today at *CIRCUIT CELLAR INK*. Because the project emphasis changed considerably in the transition and we have come to specialize mostly in embedded controls, some bright guy on the staff came up with the wonderful idea to advertise our talents as the "Ciarcia Design Works.". I'm sure if this person had realized the effect that it was going to have on his workload he may not **have been** such a proponent of advertising.

As a result, there are lots of projects going on and some of them are really wild. One project involves designing an improved method for monitoring highway speeds. As a Porsche owner, I can't for the life of me understand why I'd want to cut my own throat, but then there is the challenge of designing the unbeatable (or, better yet, knowing how it works).

I receive two or three calls a day and everyone is adamant that they need a custom-engineered solution to their problem. They see single-board computers presented for this application, special embedded controllers for that one, and so on. It's a vicious rat race to solve specific requirements by creating endless custom hardware. Since I hate make-work jobs, however, I spend about half my time on the phone trying to talk many of them out of it.

For example, yesterday I got a call from a company that needed a special custom design that could "interface to a standard ASCII modem on one side, buffer and translate messages, while simultaneously communicating with a **UART-compatible** non-ASCII serial device, in its special protocol, on the other side." Apparently I was called to see if the Design Works would do the hardware design for less than the **\$40k** quote he presently had.

You shouldn't have to be a computer engineer to realize that all this fancy specification translates to any off-the-shelf single-board controller with two serial ports and a reasonable amount of memory. In fact, I reminded him that I had presented a single-board controller, with some spooler software, that was probably exactly what he needed. It costs \$119. Next call? Be right there.

It turns out that half the calls from the Design Works ad involve telling people that they don't need custom engineering.

I don't know whether it is the fault of embedded controller manufacturers and their approach to advertising or that who new groups are becoming aware of the benefits of computer control without realizing the concept of generic electronics. There is a big crowd out there who still doesn't realize that a computer is a computer.

The idea behind embedded controllers is to provide a hardware platform for a variety of applications. Install a block of specially coded memory in Brand X controller, program it to sequentially spit out portions of this memory block through a DAC in response to specific inputs, and you have a digitized voice annunciator for an elevator. Take the same Brand X controller, change the program to have the DAC output values calculated on-the-fly from real-time analysis of specific inputs and you have an ink-jet controller for a high-speed printing press. Have the DAC control a hydraulic oil pump and it's the landing gear controller on an airplane. And so on.

Unfortunately, either many people have not evolved an understanding of a generic embedded control device where one the program is "designed," or we are forgetting that underneath all the flashy applications that there is a common ingredient: the architecture of computercontrolled devices.

While embedded controllers will never be **PALs** (programmable array logic) per se, thinking of them as logically programmed analog/digital control modules goes a long way toward educating people to think of them as controllers with a specific application program rather than a specially designed piece of hardware with a singular function. You'd be surprised at how many people find it a revelation that the part that really needs engineering is the application software.

Don't get me wrong here. I'm not complaining. Considering that I supported a Dear Abby-like computer answer column for 10 years and I'm now part of a magazine dedicated to expanding readers' knowledge on computer applications, I feel a personal obligation to help a caller solve a problem, not just sign him up as a customer. Placing an ad and talking to the people that call keeps me tuned in to that audience.

So, what can we conclude from all this? Well, if I can spend all this space to tell you that a computer is a computer, then I must not have gotten hit by lightning yet.

