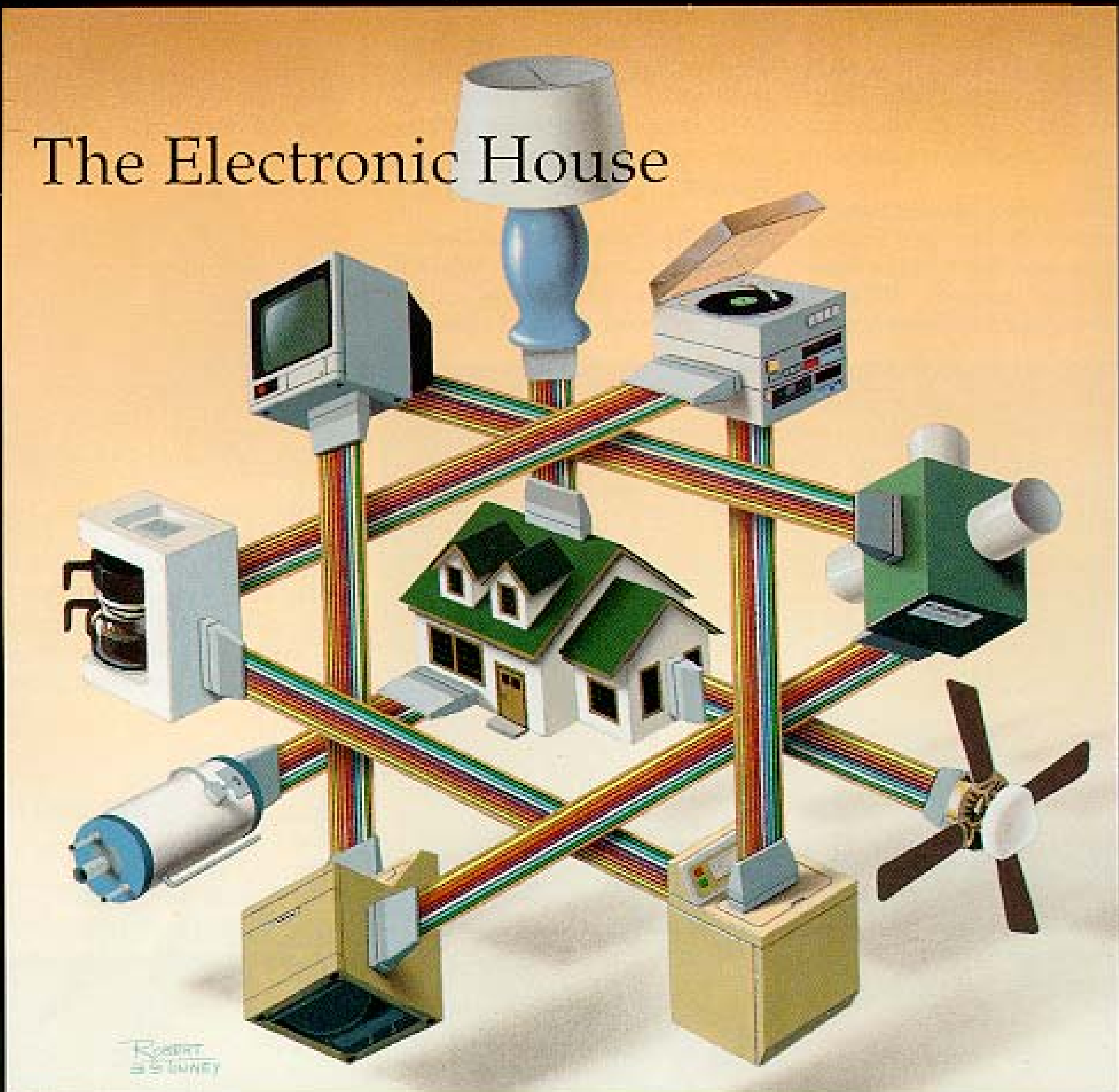
Circuit Cellar INK®

THE COMPUTER APPLICATIONS JOURNAL

The Electronic House



August/September 1989 — Issue 10

$3.95

Ctrl

EDITOR'S INK

CEBus on the Ragged Edge

Watching the beginning of an industry bears strong resemblance to watching a downhill ski race. The players move at high speed and strain to convince themselves and their spectators that they are in control of the proceedings. A champion's run lifts those who watch to dream of exhilaration and glory. A crash reminds everyone of the risks accompanying the adrenaline rush.

Building automation is not really a new industry. Companies and individuals have been automating commercial and residential buildings for over ten years now. Until very recently, though, most automated buildings have been the domains of affluent individuals or firms looking for the prestige, convenience, or very long-term overall energy savings that automation can provide. Current developments promise to open broad new markets to automation. Larger markets, and the major corporate attention they attract, can take home automation away from the exclusivity of the "early adopters" into a major industry.

As we put this issue together, we looked at the growing industry and saw signs both encouraging and dismaying. On the positive side, the technology behind CEBus, and the planning which has gone into the proposed standard before its implementation, should speed its acceptance among the general public. In addition to relatively new companies which have sprung up to work with CEBus, many of the respected names in building automation and electrical supply are looking seriously at CEBus to open new markets for their products. In the negative column CEBus is still a proposed standard. No one expects sweeping changes between the current proposal and the final specification, but cautious companies are waiting to see the final standard before committing development money.

Regardless of whether or not it ever truly takes off, CEBus is the best thing to happen in home automation in a long time. As a matter of fact, CEBus has been so beneficial to the industry, that if no CEBus products were ever released, CEBus would still be counted as a Good Thing. How can I make such an outrageous statement? I feel that CEBus has stimulated progress in the two areas where it is most lacking: public opinion and large-company interest.

The popular press has jumped on the news flowing from the CEBus camp and displayed home automation as a technology which will smooth our way into the new millennium. Whether CEBus is up to the task remains to be seen, but the public (read: potential customer) has been given the notion that an automated home is a present possibility for the average family. There are still thousands of individuals who are taxed in trying to program a VCR, so designing automation products which are easily used will be a challenge. The perceived size of the potential market will be the deciding factor in whether the challenge is worth taking up.

Potential market size will also decide who will play the game. We live in a consumer society in which brand recognition is paramount. The effect of companies such as General Electric and Matsushita entering the home automation market will be similar to that of IBM entering the personal computer market. While the big companies will capture significant market share, their presence will legitimize the field,increasing the size and value of the total market.. Major companiesareinterested in CEBus. After slogging through the VCR compatibility battlefield they know the value of a single standard to which all products attain. Whether CEBus becomes that standard or not, the companies are sharing ideas on standards which will benefit both consumers and producers. No matter what it is called, there most certainly will be a Home Automation Standard.

I am excited by much of what I've seen in the last few months, and I wish that I could write an uncompromisingly optimistic editorial on the future of home automation. There is tremendous potential, but we live in a complicated society where success for any technical endeavor is far from assured. More homes will be automated-I'm sure of that. We simply won't know the system to be used until all the racers have had a chance at the mountain.

Curtis Franklin, Jr.
Editor-in-Chief

FOUNDER/
EDITORIAL DIRECTOR
Steve Ciarcia

PUBLISHER
Daniel Rodrigues

EDITOR-in-CHIEF
Curtis Franklin, Jr.

ASSOCIATE
PUBLISHER
John Hayes

ENGINEERING STAFF
Ken Davidson
Jeff Bachiochi
Edward Nisley

CONTRIBUTING
EDITORS
Thomas Cantrell
Jack Ganssie

CONSULTING
EDITORS
Mark Dahmke
Larry Loeb

CIRCULATION
COORDINATOR
Rose Mansella

CIRCULATION
CONSULTANT
Gregory Spitzfaden

ART DIRECTOR
Tricia Dziedzinski

PRODUCTION
ASSISTANT
Lisa Hebert

BUSINESS
MANAGER
Jeannette Walters

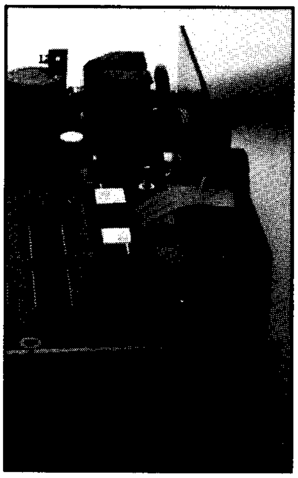
STAFF RESEARCHERS

Northeast
Eric Albert
William Curlew
Richard Sawyer
Robert Stek
Midwest
Jon Elson
Tim McDonough
West Coast
Frank Kuechmann
Mark Voorhees

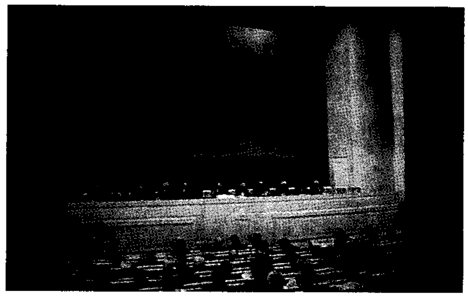
Cover Illustration
by Robert Tinney

CIRCUIT CELLAR INK

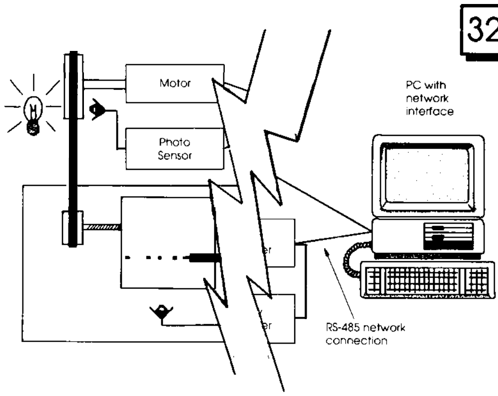
FEATURES

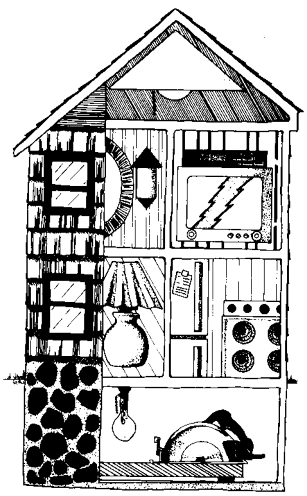


The Soviet Union uses an impressive array of broadcast satellites to send television throughout their country. If you live in North America you can track and receive from these satellites using Mark Dahmke's techniques.



DEPARTMENTS





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

CIRCUIT CELLAR INK (ISSN 08968985) is published bimonthly by Circuit Cellar Incorporated, 4 Park Street, Suite 20, Vernon, CT 36066 (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 $ 17.95, all other countries $26.95. 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 2099, Mahopac. NY 10541 or call (203) 875-2 199.

POSTMASTER: Please send address changes to Circuit Cellar INK, Circulation Dept., P.O. Box 2099, Mahopac. NY 10541.

Ctrl

READER'S INK Letters to the Editor

NEURAL NETWORKS

Let me say that the neural network articles were fantastic! In my opinion, [Issue #9] was your best issue yet (they keep getting better). I know that it might not be easy to get regular articles about AI and neural nets, but if possible feel free to get another one printed.

Would it be possible to include additional reference books at the end of an article as a sort of "For more information..."? I think that could help guide readers in a specific direction if they are interested in the topic discussed.

Mark Balch
(From the Circuit Cellar BBS)

We are planning to publish more articles concerning artificial intelligence. There are a number of problems that can be solved using AI techniques, and we plan to present these powerful tools to our readers.

You're right, a section coveringfurther information would be useful on many of our articles, and we plan to include more of this type of information in the future. To get started, check out "Experiment in Artificial Neural Networks" by Ed Rietman, published in 1988 by Tab Books (ISBN O-8306-9337-8). This book covers neural networks from a hardware point of view, and includes both theoretical and hands-on sections. ***Editor.***

I have a suggestion for CCINK. Consider a column or series on the engineering process itself. Something like "Advice from the Oracle." Topics could include: 1) Selecting a CAD system-what features to look for, what "features" to avoid, etc. 2) Using a CAD system-techniques, tips, and tricks. 3) Designing and fabricating a PCB. 4) Selecting various test equipment. 5) Various hardware design topics (e.g., designing a high-speed memory system or designing a basic text video display). Such a series would benefit a wide range of readers from beginners to seasoned engineers.

P.S. The article on neural nets was great. I've heard a lot of babbling about neural nets in other periodicals and books, but Chris Ciarcia's article clearly explained how they worked, not just what they could do.

James O'Sullivan
(From the Circuit Cellar BBS)

Thanks for the suggestions. We have been looking at various ways of presenting information such as you suggest in a way that won't duplicate the articles available in other magazines. Other readers have given us suggestions, and we're always happy to hear about ways we can make CIRCUIT CELLAR INK fit readers' needs more completely.

Our goal is to present articles that help readers with the challenges they face in the real world-not articles on theo y, but on how to apply theory. We're glad that the neural network articles worked in that catego y.

MORE ON LCDS

In the enjoyable LCD article (CIRCUIT CELLAR INK #8) by Ed Nisley, he mentions the Hitachi H2570 1 x 16 units available from Timeline. However, as the Timeline ad in that issue shows, they are now selling a similar unit, the DMC16106A, from Optrex, at the same price--$10. Like the unit mentioned in the article, the Optrex module is based on the Hitachi HD44780 and even has the same connector pinout.

The major difference is that the newly offered Optrex unit includeselectroluminescent backlighting. The 'backlight!' consists of a thin (1.5mm) layer of organic material underneath the LCD which emits a bluish-colored light, providing high contrast for the black characters. This is great, since LCDs suffer from notoriously bad readability in all but direct light.

I ordered the Optrex and tried it out. Connecting the 14-pin edge connector as described in the article produced characters on the display but no backlight. Unfortunately, the data sheet I ordered along with the Optrex LCDs had absolutely nothing to say about backlighting, so I checked around. Here is the story:

It turns out that the EL panel, which has two obvious (though unmarked) solder **pads** for power, expects 100–150 VAC. Now, before anyone gets hurt trying to plug their LCD into a wall socket, note that the voltage isn't as important as the frequency. Notably, the 60 Hz in the wall will generate a very dim light. Instead, you need somewhere around 400-600 Hz!

Luckily, power inverters especially designed for backlight applications **are** available. I found some (Optrex D32-45) listed in the Digi-Key catalog (part number OP002, $9.50 each) that convert the 5 VDC to 100 VAC at 400 Hz. For battery operations, only a few milliamps of 5 VDC are required to "lighten up."

T.W. Cantrell
Fremont, CA
(Tom Cantrell is a Contributing Editor of CIRCUIT CELLAR *INK.)*

THE MOPOKE, RIP

You nitwits! You claim to be impartial and objective. I know why you didn't include the Sunbeam 101 in your lousy review. My Sunbeam can toast bagels around your crumby P-S TPB-5342.

Eight brown level settings. Phooey! The Sunbeam 101 has infinite proportional control for toasting range. The full 0% to 100%. And what about service? The Sunbeam has a hinged bottom for immediate access (not that one ever needs it) to all of the precision inner parts.

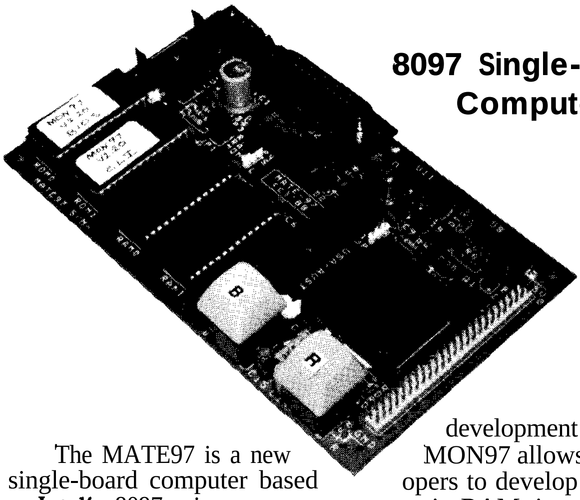
Handsome chrome case. Ha. The Sunbeam has a copper clad steel case finished off in heavy chrome plate meeting all current and future FCC requirements for EM1 suppression. It also works equally well on either AC or DC, and is immune to voltage transients and poor regulation. I'll bet your TPB-5342 can't top that.

D.S. Jenkins
Tarzana, CA

You're right. We were obviously guilty of being short sighted, narrow minded, and overfly prejudiced. You have uncovered our plot to deny the Sunbeam the market if so richly deserves.

Now that our plot is discovered, there is but one honorable course of action: The Lab Staff will be taken out and shot on the morrow.

We hope you're happy now. Editor.



8097 Single-board Computer

The MATE97 is a new single-board computer based on Intel's 8097 microcontroller. The MATE97 is a 6.1" x 3.6" controller that provides up to 32K of static RAM, 16K or 32K EPROM, and a 12-MHz 8097 along with a single 50-pin header for I/O. With its simple I/O interfacing and thoroughly tested hardware, the MATE97 has been designed to provide a solid foundation for hardware development and software prototyping.

The MATE97 Developers Kit includes the MATE97 computer and MON97 system monitor firmware. With the addition of a terminal (or PC with terminal emulation software), the engineer has a complete development system. MON97 allows developers to develop programs in RAM, interrogate I/0 ports, and debug code with breakpoints and single stepping. If large projects require more programming, an in-line assembler, cross-assembler, and C compiler are available for development on IBM PCs.

MATE97 Developer Starter Kits are available for $495.

The C96 cross-assembler and C compiler for the PC are available for $250 each.

B. G. Associates, Inc.
P. 0. Box 66
Bowie, MD 20715
(301) 262-5567

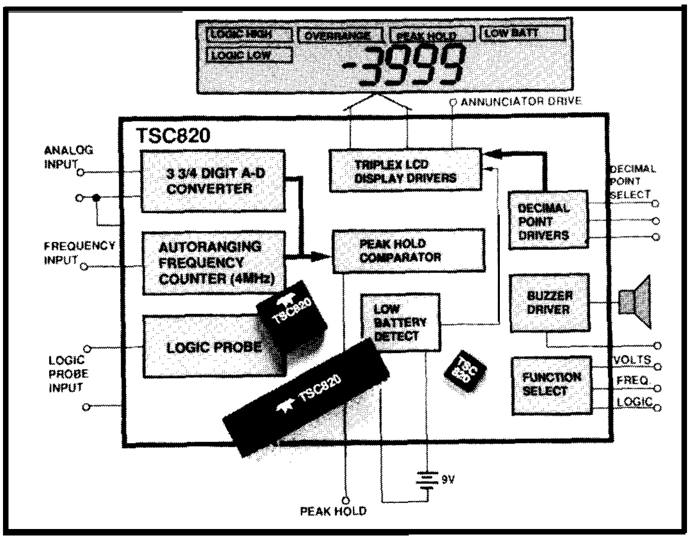
Data at 100 MHz

Demanding applications have forced designers and engineers to sample data at ever higher frequencies and bandwidths. In response to the needs of applications such as spectrum analysis, video processing, radar, and image processing, Signatec has released the DASPlOO, a single-board 100-MHz data acquisition and signal processing system. The DASPlOO is a standard ISA bus board designed to work in any AT-bus-compatible computer with EGA graphics adapter.

The DASPlOO samples data at 100 megasamples per second at 8 bits of resolution. The board has 256K of signal memory, which can be expanded to 2 megabytes, and up to 32K of processor memory. The user may select between a number of operation and control modes including switched-gain linear amplifier and wide-range compression amplifier, and continuous, pretrigger, and burst data acquisition modes.

The DASPlOO is designed for use in single- or multi-processor applications with up to three boards residing in a single host computer. The DASP100 uses the TMS320C25 DSP, with a peak rate of 10 MIPS. DASPlOO systems are available starting at $5780, with delivery from in stock to 30 days.

Signatec, Inc.
357 Sheridan St., Suite 119
Corona, CA 91720
(714) 734-3001



Single-Chip Measurement System

A single-chip device combining an analog-to-digital converter, a logic probe, a frequency counter, and an LCD driver has been introduced by Teledyne Semiconductor. The TSC820 has a resolution and display driver of 3 and 3/4 digits (maximum count 3999), and includes functions for decimal point drivers, low-battery detector, buzzer output driver, and peak reading hold along with its measuring and conversion capabilities.

The TSC820's A/D converter provides guaranteed zero reading and a zero-integrator cycle as well as a 100-µV resolution up to 400 mV. The internal voltage reference has a typical drift of 35 ppm "C.

The frequency counter is autoranging over four decades and operates at up to 4 MHz with accuracy limited by the host circuit's oscillator crystal.

The logic probe inputs of the TSC820 are tied to external level shifter permitting adjustment for most logic families. The chip signals logic level low by turning on a buzzer.

The peak reading hold retains the highest reading from the A/D converter or frequency counter. The digital display register value will hold a true reading over time, immune from the bias current leakage that can cause "droop" in analog peak-reading circuits.

The TSC820 is available in 40-pin DIP, 44-pin PLCC, and 44-pin plastic flat packages. All packages are certified for both commercial (0°C–70°C) and extended (-40°C–85°C) temperature ranges.

Pricing for the TSC820 begins at $6.00, quantity 100.

Teledyne Semiconductor
1300 Terra Bella Avenue
Mountain View, CA
94039-7267
(415) 968-9241

New Cross-Development Software for the Macintosh

Embedded systems developers using Macintosh development systems now have the same powerful array of high-performance cross-development tools that have long been available on other popular development systems. INTROL recently ported its entire line of INTROL-C Cross-Compiler

Cross-

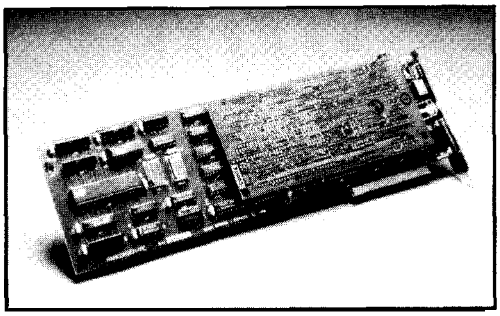
INTROL's

a full-time cost-effective software development environment.

Target processors supported by INTROL's family of cross-development tools include the 6301/6303, 6801/6803, 6804, 6805, 6809, 6811C11, 68000/68010, 68020/68030,

Milwaukee, WI 53204
(414) 276-2937

NTSC/PAL Input Frame Grabber Board for IBM PC/XT/AT/Compatibles



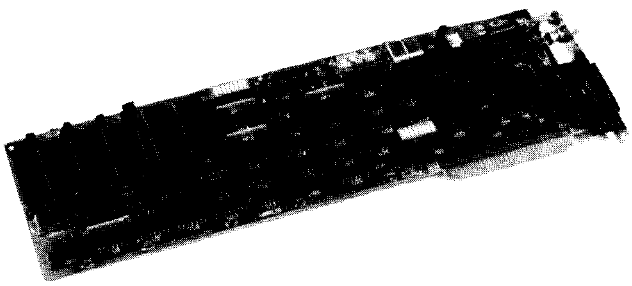
Low-cost, real-time image capture from NTSC, PAL, VTR, or VCR sources is possible with MetraByte's MV-NTSC Frame Grabber Board. It performs all analog signal conditioning and synchronization in producing a digitized image which can be viewed on a standard VGA or multisync analog monitor in the 320 x 200, 256-color mode. Monochrome monitors with 16 gray levels and 640 x 480 resolution are also supported.

The board was specifically designed to economically digitize three-dimensional objects or documents too large for conventional desktop scanners. Live video images can be displayed at a three-frame/second rate on a standard color TV/monitor or multisync monitor connected to the MV-NTSC as a secondary display. Real-time images are captured at a 1/30-second acquisition time.

The $999 board includes menu-driven software to control the operation of all features including: image acquisition; image storage and recall; board set up, configuration, and control. Programs that convert a captured image to popular desktop publishing and graphics formats such as PageMaker, Ventura publisher, **Dr. Halo, and PC Paintbrush, are also included.**

MetraByte Corporation
440 Miles Standish Blvd.
Taunton, MA 02780
(508) 880-3000

Low-Cost 68008 Multiprocessor for PC/AT



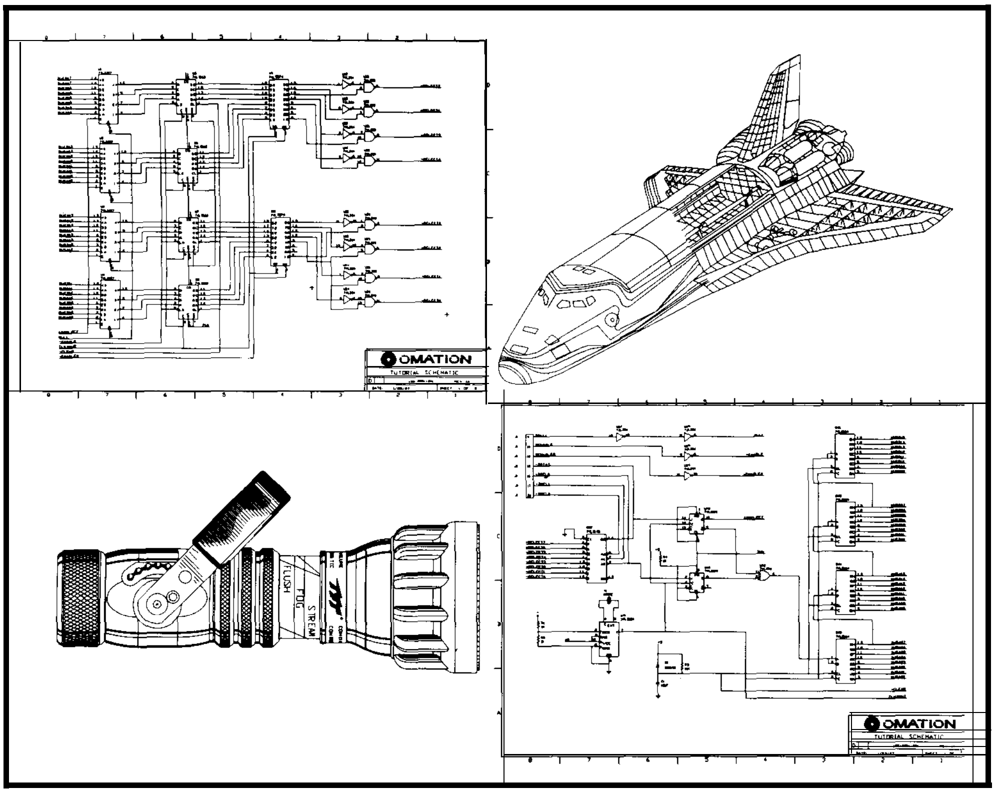
A complete 68000 based computer system that plugs into a PC, AT, or compatible has been announced by Micro Resources. The MISTER-8 is a full-length board that uses a 68008 microprocessor with an 8-MHz clock. The 68008 is code compatible with the 16/32-bit 68000, but has an 8-bit external data bus. The board contains 64K of static RAM, provision for 128K of ROM, a monitor EPROM, and a Zilog 8038 128-byte FIFO buffer to the PC's I/O channel. A serial port is also provided. Development time is improved since users get instant feedback when running and debugging programs with Motorola's TUTOR monitor. Complex projects can be simplified by using the familiar PC editor and cross-assembler tools to write and document programs. The MISTER-8 monitor contains system-call routines to allow quick development in C or assembly language. It also supports Ready Systems VRTX32 real-time executive and can be used to develop embedded multitasking applications.

Two or more MISTER-8 boards can be installed in a PC to develop parallel-processing techniques and programs. Operator control is through a FIFO buffer to allow fast transfer of code and data from the PC. Programs can be downloaded to each individual board, or optional application programs can be installed directly on the 68008 board. The boards can also be used as intelligent, dedicated instruments within the PC. Its serial port, for example, can be connected to a variety of remote devices for measurement and control.

The $345 board is based on the book "68000 Microcomputer Systems: Designing and Troubleshooting," by Alan D. Wilcox, Prentice-Hall, 1987.

Micro Resources **Company**
60 South Eighth Street
Lewisburg, PA **17837**
(717) 524-7390

linking CAD and Desktop Publishing



Omation has released the newest version of SCHEMA, a schematic capture program for electrical engineers. The new version, SCHEMA II+ version 2.2, allows users to export files to other CAD programs and work with many of today's popular desktop publishing systems.

The most noticeable changes to SCHEMA in the newest version are the additions to its roster of output formats. In addition to the plotters and printers traditionally supported by CAD packages, SCHEMA II+ 2.2 will serve up output in Postscript, TIFF, DXF, WKS, DBF, SDF, PCX, Delimited, and PSpice format files. The combination of output formats allows SCHEMA II+ 2.2 to provide electrical engineering CAD drawings for inclusion in documents prepared by PageMaker, PC Paint, Publisher's Paintbrush, Ventura Publisher, and other major DTP and paint programs.

Engineers from offices with multiple CAD packages may be more impressed with SCHEMA II+ 2.2's ability to exchange complex files with AutoCAD. Unlike many DXF exchange features which only work to transfer relatively simple files, the DXF transfer feature of SCHEMA II+ 2.2 is designed to include blocks, polylines, attributes, hidden attributes, arcs, circles, and other advanced shapes and segments.

SCHEMA II+ 2.2 runs on PC, PC/AT, 80386, PS/2, and PC-compatible computers. The list price of the package is $495 with same day shipment and 30-day money back guarantee. A free evaluation kit and demo disk are available upon request.

Omation, Inc.
801 Presidential Drive
Richardson, TX 75081
(800) 553-9199 or
(214) 231-5167

Micro Channel Interface on a Chip

Designing an interface product for the IBM PS/2 Micro Channel bus has been greatly simplified with ONE CHIP PLUS, the 88C01 Micro Channel Interface Chip from Capital Equipment Corp. It provides complete Micro Channel interface compatibility plus the decoding and logic functions required for memory, I/O, and multifunction adapter cards.

Features of the 84-pin PLCC chip include Direct Memory Access (DMA) arbitration and burst mode DMA, programmable memory and I/O timing to accommodate wide access times on memory and peripheral chips, user-definable Programmable Option Select (POS) registers, and a programmable Micro Channel board ID. In addition, multiplexed memory address lines allow direct connection to 1M-bit DRAMs.

A unique user configuration capability permits adjusting the characteristics and pinout of the chip to specific design needs. Some of these characteristics include: enabling expanded and extended memory, data width for each chip select, I/O access time, ROM size, and Micro Channel ID number. Over 20 user configuration features can be set with a configuration PROM.

A development kit provides complete support for the Micro Channel interface design and includes a prototype board with a working memory adapter circuit, breadboarding area, and connections for a logic analyzer. Interactive 88C01 configuration software, EPROM-based code for memory add-in cards, design utilities, and a DOS expanded memory driver meeting EMM 4.0 are also provided.

Acquisition Engine software, which provides background or foreground data acquisition, general-purpose I/O control, and graphics is also included. Customized software modules can be added for specific products.

Documentation includes specifications, schematics, application notes, recommended PC layout, and CAE parts libraries.

The entry-level development kit is $495 with a 30% discount to CIRCUIT CELLAR INK readers.

Capital Equipment Corporation
99 South Bedford Street
Burlington, **MA 01803**
(617) 273-1818

Science, Engineering & Graphics Tools for

MS C, MS Quick C, MS Fortran*, MS QuickBasic, Turbo C, Turbo Pascal

The Science/Engineering/Graphics Tools 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 150 page manual describes the form, function, and parameters of each procedure and function. Theses tools are available for Turbo Pascal 4.0, 5.x, Turbo C 1.5, 2.x, Microsoft C 5.x and QuickC, QuickBasic 4.x and Microsoft Fortran 5.0 for IBM compatibles.

Ordering Information

Model#	Version	Price
IPC-TP-016	Turbo Pascal 4.0, 5.x	$ 79.95
IPC-TC-006	Turbo C V 2.x	$ 79.95
IPC-MC-006	Microsoft 5.1 & Quick C	$ 79.95
IPC-QB-006	QuickBasic V 4.x	$ 79.95
IPC-MF-006	Microsoft Fortran V 5.0'	$150.00

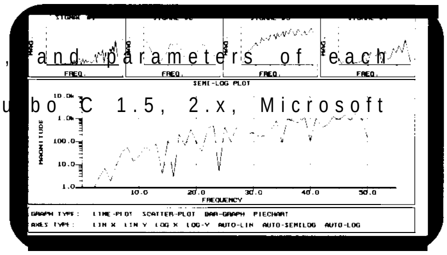
Shipping charge is $3 00 within USA Elsewhere add $18 00 for shipping Mastercard. Visa, Company PO's , and personal checks accepted MASS residents add 5% sales tax

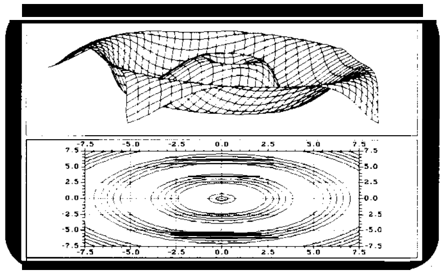
*Microsoft **Fortran** 5.0 Version Available in September*

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)

Hardcopy support Epson MX, FX and LQ printers, HP plotters, HP Laserjet and Thinkjet printers, Toshiba 24 pin printers and other devices

Science/Engineering charting routines Linear, semi-log, and log graphs. Auto-scaling of axes, line, scatter, pie, and bar charts. *Charting Graphics Now **Includes** Contour Plotting.*





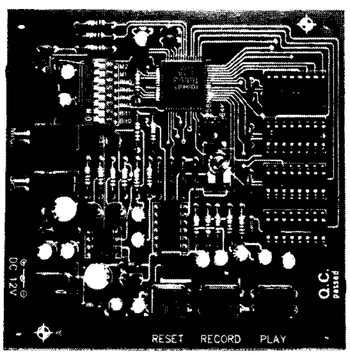
3-D plotting translation, scaling, rotation, and perspective routines

Statistics - mean, mode, standard deviation, standard error, etc.
Multiple Regression - With summary statistics
Curve Fitting - Polynomial and cubic splines
Simultaneous Equations real and complex
Fourier Analysis Forward and inverse FFT, Rectangular, Parzen, Hanning, Welch, Hamming, and Exact Blackman Windows, 2-Dimensional FFT and Power Spectrum
Matrix Math Real and complex
Complex Number Arithmetic
Eigen values and vectors
Integration - Simpson's method
Differential Equation - Runge-Kutta-Fehlberg
Linear Programming - Simplex method
Root Solving Bisection, Newton and Brent methods
Data Smoothing
Special Functions Gamma, Beta, Bessel, error, hyperbolic trig, orthagonal polynomials
RS-232 Support all versions include an interrupt driven RS-232 driver

Quinn-Curtis

1191 Chestnut St., Unit 2-5, Newton, MA 02164 USA
Tel. (617)965-5660 FAX (617)965-7117

Circle No. 145 on Reader Service Card



— DIGITAL VOICE MODULE —

- Low cost
- Selectable banks
- 4 Sampling rate
- Super quality
- 1 W amp
- DRAM operation

DVM-1 — $49 (without RAM)

— PASSIVE INFRARED DETECTOR —

- Used in alarm system, moving detection
- Super sensitive
- Very reliable
- Exchangeable lens
- Analog Pulse Count

RK4000PCA — $59

MING *Engineering, Inc.*

515 S. Palm Ave., #5
Alhambra, CA 91803
(818) 570-0058 **Fax: (818) 576-8748**

Circle No. 141 on Reader Service Card

R&D ELECTRONIC SURPLUS, INC.

Has been in business since 1946 selling NEW surplus electronics and electromechanical parts.

Send for our FREE 40 page catalogue detailing:

Batteries	LEDs
Cables	Lugs
Capacitors	MOVs
Clocks	Ni-Cads
Connectors	Power Devices
Digital Timer/Controllers	many Power Supplies
Diodes	Relays
Displays	SemiConductors galore
Enclosures	Stepper Motors & Driver ICs
Fans	Speakers
Filters	Many Switches
Heat-Shrinkable tubing	Telephones and components
Heatsinks	Transformers
Integrated Circuits	Zeners
Lamps & Lights	etc.

R&D ELECTRONICS

1224 Prospect Avenue
Playhouse Square
Cleveland, OH 44115

Telephone: (216) 621-1121 • Fax: (216) 621-8628

Circle No. 146 on Reader Service Card

Ctrl

VISIBLE INK

Letters to the INK Research Staff

Answers; Clear and Simple

EPROM/PAL PROGRAMMER ON THE CHEAP

I've been looking for an EPROM/PAL programmer pin-driver circuit for a while now without much luck. I think Elantec offers a monolithic one, but they're expensive if you want to buy 40! Do you know of a reasonably compact circuit which will handle programmable voltage, slew, sink, source,and tristatenecessary for a good-quality programmer?

Ralph Ursoleo
Eugene, OR

EPROM/PAL programmer designs conform to the old cliche that "there's more than one way to skin a cat." Because of the differing requirements of EPROMs and various PALs, truly "universal" designs are all but impossible. "Dumb" serial interfaces, such as the UART-based design described in the Februa y '85 "Ciarcia's Circuit Cellar" (in BYTE, also in Volume VI of the collected articles, available from both CCI and Micromint) represent one extreme. This type of design is based on easy-to-find components readily found from sources like JDR Microdevices, Jameco Electronics, and traditional industrial suppliers like Hamilton-Avnet. The later 8052-based SEP is a "smart," upgraded implementation of the same basic idea. Both designs are specifically tailored to the programming needs of EPROMs rather than PALs and cannot be readily modified to accommodate the latter.

The article titled "A PAL Programmer" by Robert A. Freedman in the Janua y 1987 issue of BYTE gives details for a design based on the Sprague UCN5810A, UCN5821A, and UCN5895A chips. None of these chips does everything needed, but in combination with the rest of the circuitry in the design, they together can be made to do everything necessa y for programming PALs. Making modifications to Freedman's design to use a processor like an 8052 instead of a host PC would give asmarterstand-aloneunit. These Sprague chips are probably the closest you'llget to reasonablypriced components that meet your needs.

PROGRAMMABLE THERMOSTAT SEARCH

Is there a relatively inexpensive digital thermostat that would provide true binary-data feedback to a controller and thus to a central HVAC unit? I feel certain that such a unit must be available off-the-shelf (probably manufactured by Johnson, Honeywell, or Barber-Coleman) but I am unable to locate or even identify one. I should think it would be thermistor based, but I can't see that I should have to design the thing myself if one is available commercially.

Michael Newell
Arlington, TX

Industrial "thermostat" temperature monitor/controllers are available in many types and sizes (check out the current Omega Engineering "Temperature Measurement Handbook and Encyclopedia" for some of the possibilities) with sensors ranging from thermistors to thermocouples to platinum RTDs. Some consist mostly of a heavy case with a minimum of delicate components, others use microprocessors for the "brains." Older mechanical types are being rapidly and more cost-effectively supplanted by systems using analog-to-digital converters with solid-state temperaturesensors and op-ampsignal conditioners. An op-amp, a thermistor or LM335, a potentiometer, and a few additional componentssuchasanoutput relay,can becombined for a ve y simple, relatively accurate thermostat with a variable switching point (set point) that is much more cost-effective and repeatable than the bimetallic type it can replace. Thesolid-state sensors are currently more cost-effective than most other types, including thermistors, for home control applications. The National Semiconductor LM334 is a popular sensor, but there are others such as the LM335, LM34, LM35, and the Analog Devices AD590 (second-sourced by Intersil as the AD590I). An ordina y transistor or diode can be used as a temperaturesensor whengreatprecision isn't needed; theforward voltageofasilicon diodevaries quite linearly with temperature, especially with the very limited temperature ranges usually encountered in home con trol applications.

The most practical way to convert a continuously vaying (analog) quantity to something a digital computer understands is with a device called, appropriately, an "analog-to-digital converter." While they can be built upfrom discretecomponents or integrated circuits (e.g., comparitors) and resistor ladders, there are a number of *low-cost "successive approximation" converters availablefrom manufacturers such as National Semiconductor that incorporate nearly everything in a single IC package. A device like the ADC0809 provides eight independent conversion channels very inexpensively and without a lot* of *bother.*

The "Ciarcia's Circuit Cellar" article "A Computer-Controlled Wood Stove" (iin the February 1980 issue of *BYTE andVolume II of the* Circuit Cellar reprints,) shows the*temperaturesensor (LM334) and op-ampsignal-conditioner (LM301A) circuits, together with a National ADC0809 a-channel converter. The "output" side* of *the control loop can be handled* various ways, *determined* by the action *desired.* If *you're wiring into an existing contact-closure-type system, you may need only parallel relay-typecontacts. Ifyou'restartingfromscratch, you might want to use AC circuits switched with triacs. The many past "Ciarcia's Circuit Cellar" articles on home* **control** *illustratesomeof the possibilities (Volume VI of thecollected Circuit Cellar articles contains three of the later home control articles, followed byaseparatearticleonsensorsand output devices; also in* the *April, May, June, and* July 1985 *issues* of BYTE).

Several books have been published describing home control systems run by everything from Apple][and IBM PC-type computers tosimplestand-alone6502systems. A good technical library and a bookstore are good places to track down some of these. Titles such as Blankenship's "The Apple House," Auslander and Sagues' "Microprocessors for Measurement and Control" (Osborne/McGraw-Hill), Wobschall's "Circuit Design for Electronic Instrumentation" (McGraw-Hill), and Carr's "Digital Interfacing with an Analog World" (Tab), come to mind. There are many others.

In Visible INK, the Circuit Cellar Research Staff answers microcomputing questions from the readership. The representative questions are published each month as space permits. Send your inquiries to:

INK Research **Staff**
c/o Circuit Cellar INK
Box 772
Vernon, CT 06066

All letters and photos become the property of **CCINK** and cannot be returned.

IRS

201 Very Useful
202 Moderately Useful
203 Not Useful

An Intelligent SCSI Data Acquisition System for the Apple Macintosh

PART 2

by John Eng

Macintosh Programming The Easy Way

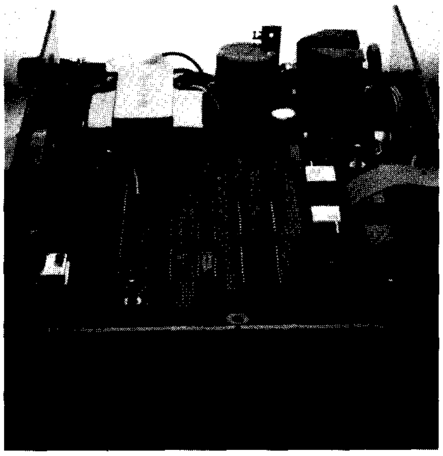
In the feud between IBM PC/AT and Apple Macintosh proponents, the Macintosh side has been disadvantaged by the notion that the Macintosh is extremely difficult to program. Generally speaking, this notion is not true. Standard implementations of a varietyofprogramminglanguagesare available for the Macintosh. Therefore, writing a Turbo Pascal program on the Macintosh should be no more difficult than writing the same program using Turbo Pascal on a PC.

Macintosh programming becomes substantially more difficult if the programmer decides to implement a mouse-driven, Macintosh-style user interface instead of a conventional TTY-style, keyboard-oriented interface. With menus, windows, and buttons, a well-written Macintosh-style user interface provides more intuitive control over the application program than does the TTY method. Since there are no program subsections to enter before performing a specific action, the user interface is said to be modeless. Not surprisingly, the flexibility and graphical appearance of a Macintosh-style user interface require a significant amount of additional program code.

Although Macintosh purists may disagree, the programcode to provide a Macintosh-style user interface is entirely optional. In fact, most Macintosh compilers provide standard I/O procedures for a TTY interface to maintain compatibility with standard language definitions. For a Macintosh hardware designer who needs a program to test a new project, it is reasonable to ignore the Macintosh user interface and write the driver program using the standard TTY I/O procedures supplied with the compiler. As the program becomes a full-scale Macintosh application, the programmer may then put in the extras for the Macintosh-style user interface.

There are a number of software tools available for implementing a Macintosh-style user interface. At the lowest software level, the Macintosh ROM and operating system provide a wide variety of system-level subroutines known collectively as the Macintosh Toolbox. Toolbox routines are callable from most high-level languages, and such calls are supported by most Macintosh compilers. In addition to an extensive collection of routines related to the Macintosh user interface, the Toolbox contains routines for memory management, device drivers, sound synthesis, and more. (The complete Toolbox documentation covers over four volumes.)

Even with the Toolbox, implementing a Macintosh user interface is time consuming at best. A number of high-level software tools areavailable to simplify the creation of Macintosh userinterfaces: most notable are Prototyper fromSmethersBarnes, Programmer's Extender from Invention Software, and MacApp from Apple Compu ter. Prototyper generates high-level source code for a Macintosh-style user interface according to a programmer's specification. In contrast to Prototyper, which is an application, Programmef s Extender and MacApp are high-level subroutine libraries. Programmer's Extender subroutines are callable from a number of Macintosh compilers, while MacApp requires the Macintosh Programmer's Workshop,



Apple's exhaustive programming environment for the Macintosh.

SCSI AND THE DAQ3000

The DAQ3000 SCSI Data Acquisition Device relies heavily on the SCSI interface to achieve its 28-kHz sampling rate. The SCSI standard, as defined by the American National Standards Institute (ANSI), specifies the sequence and timing of 18 TTL-level signal lines. Figure 1 is a schematic diagram of SCSI bus timing. In SCSI terminology, the Macintosh always acts as the initiator device, while the DAQ3000 operates as a target device. Other SCSI devices on the bus, such as hard disks, also operate as target devices. Communication over the SCSI bus is divided into several bus phases. During the first phase, the selection phase, the Macintosh (as the initiator) asserts SEL\ and DB*n*\, where *n* is the DAQ3000's SCSI target ID number (0–7). The current DAQ3000 software assumes a target ID number of 4. When the DAQ3000 detects the SEL\ signal along with the appropriate DB*n*\ signal, it responds by asserting BSY \. The Macintosh

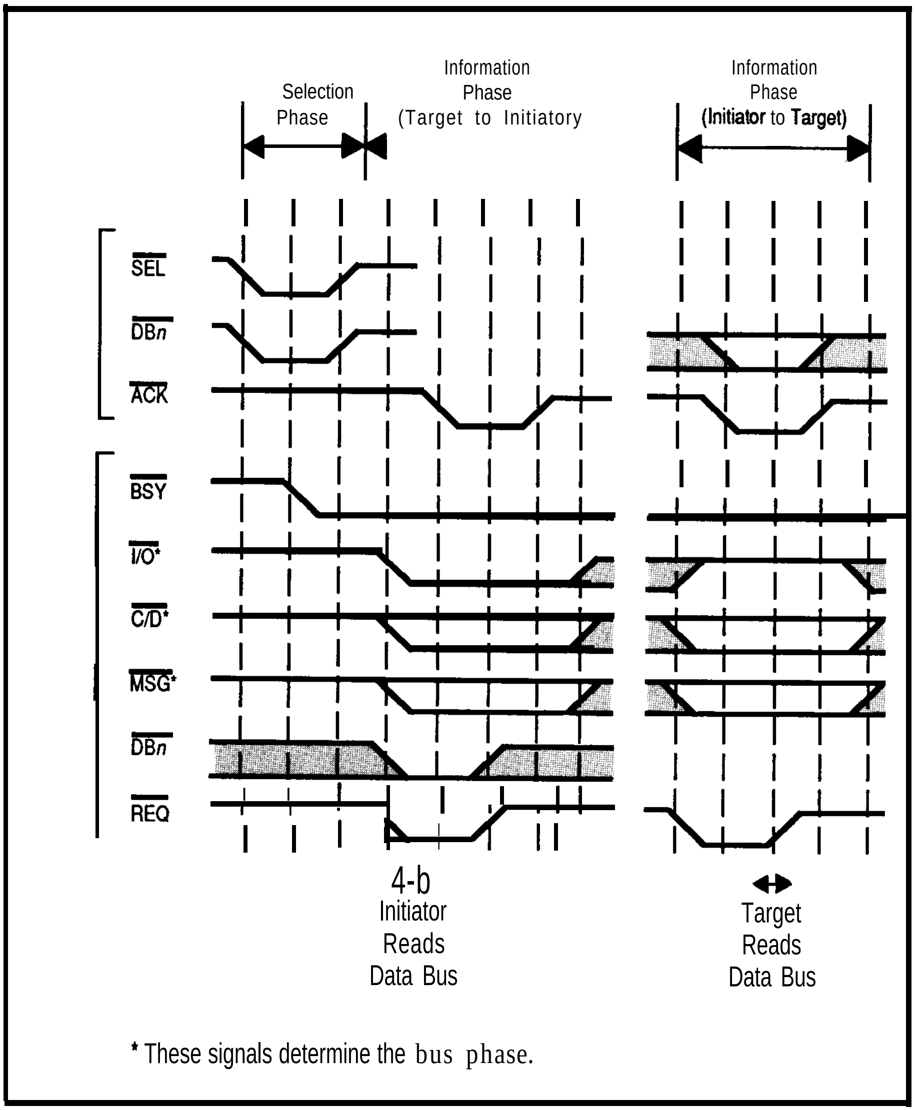


Figure 1 *—SCSI transfer protocols*

repliesbydeassertingSEL\ andDBn\, completing the selection phase.

The selection phase is followed by information transfer phases. During the information transfer phases, the DAQ3000 has control of the SCSI bus until it deasserts BSY\. For all information transfer phases, the REQ\ line (driven by the DAQ3000) and the ACK\ line (driven by the Macintosh) are used for handshaking. To transfer a byte of data to the Macintosh, the DAQ3000 asserts the appropriate eight data bits (in DB0\–DB7\) and then asserts REQ\. The Macintosh replies by reading the eight data lines and asserting ACK\. The DAQ3000 then deasserts REQ\ and the Macintosh reacts by deasserting ACK\. To obtain a byte from the Macintosh, the DAQ3000 asserts REQ\. The Macintosh replies by asserting the appropriate eight data bits (in DB0\–DB7\) and then asserting ACK\. The DAQ3000 responds by reading the eightdatalinesanddeassertingREQ\, and the Macintosh replies by deasserting ACK\. Within an information transfer phase, the REQ\ and ACK\ handshaking cycle is repeated until all data bytes are transferred. The type of information transfer phase determines the direction of data transfer and the general meaning of the transferred data. The DAQ3000 sets the type of information transfer phase by (de)asserting the I/O\, C/D\, and MSG\ signals as shown in Table 1.

The DAQ3000 supports the command, data-in, data-out, status, and message-in information transfer phases. In the command phase, the DAQ3000 receives a command descriptorblockfrom theMacintosh. The command descriptor block tells the DAQ3000 what function to perform. The DAQ3000 recognizes the four mandatory SCSI commands: read, write, request sense, and format unit. Figure 2 shows the command descriptor blocks expected by the DAQ3000 for these four commands. Byte 0 of each command isa standard SCSI command code. The significance of the remaining five bytes in each command descriptor block varies. With the read andwritecommanddescriptorblocks, a standard SCSI hard disk drive would interpret bytes l-3 as a logical block address (i.e., the address of a sector on the disk). Since it's not a block-oriented device, the DAQ3000 interprets bytes l-3 as the actual number of 16-bit words to transfer.

Upon receiving a read command descriptor block from the Macintosh, the DAQ3000 asserts the data-in phase and enters a program loop to read the ADC for the number of times specified in the command descriptor block. As each 12-bit value is read from the ADC, it is sent as a two-byte word to theMacintosh. Whenreceivinga write command descriptor block, the DAQ3000 asserts the data-out phase and enters a program loop to receive the specified number of 12-bit values (as two-byte words) from the Macintosh. As it is received, each 12-bit value is sent to the DAC. For a request sense command, the DAQ3000 sends four bytes of zeros as the sense data. The DAQ3000 also recognizes the format unit command, but does nothing on receiving this command except for returning an error condition code during the status phase (see below).

After completing the action specified by the **command** descriptor block, the DAQ3000 asserts the status phase and sends a one-byte condition code to the Macintosh. The DAQ3000 then assertsthemessage-inphaseandsends a completion byte ($00). Finally, BSY \ is deasserted, and the SCSI bus becomes free for another selection phase.

The DAQ3000 software uses the read and write commands to handle a few special cases. At boot time, the DAQ3000 mimics a standard block-oriented SCSI device because the Macintosh looks for device driver and partition information in blocks 0 and 1 on every SCSI device connected to the system. Since it is not a hard disk, the DAQ3000 sends a block of 256 zeros when it receives a read command in which the three-byte length parameter is 0 or 1. The other special case occurs for a write command in which the three-byte length parameter is 0. In this case, the subsequent SCSI data is assumed to be 6502 machine code and is read into the DAQ3000's RAM instead of being sent to the DAC. After completing the subsequent message-in phase, the DAQ3000's 6502 microprocessor calls the downloaded machine code as a subroutine. This feature is useful for running short test routines on the DAQ3000 without having to reprogram its EEPROM.

Phase	I/O	C/D	MSG
Data-Out	deassert	deassert	deassert
Data-In	assert	deassert	deassert
Command	deassert	assert	deassert
Status	assert	assert	deassert
Unspecified	deassert	deassert	assert
Unspecified	assert	deassert	assert
Message-Out	deasserl	assert	assert
Message-In	assert	assert	assert

Table 1 - SCSI bus phase signals are active-low, so 'assertion' of a signal drives the corresponding SCSI bus line to 0 volts. The DAQ3000 does not support the optional message-out phase.

THE DAQ3000 MONITOR

The DAQ3000's EEPROM contains a monitor, DAQ3000.ASM, which initializes the DAQ3000 unit, interprets the SCSI command descriptor blocks, and performs all of the DAQ3000's functions as a SCSI target device. The DAQ3000 monitor is a 6502 assembly language program developed on an Apple II+ using Apple's Toolkit Assembler. The monitor reads and asserts SCSI signals through control registers in the NCR 5380 SCSI controller chip. During the information transfer phases, REQ\ and ACK\ handshaking is handled by one of two 5380 data transfer modes. In programmed I/O mode, the monitor detects ACK\ and asserts REQ\ by explicitly reading and writing the appropriate 5380 control registers. In pseudo-DMA mode, REQ\ and ACK\ handshaking is handled automatically by the 5380 controller, and the chip behaves more like a memory-mapped parallel port. The DAQ3000 uses the faster pseudo-DMA mode for the data-in and data-out phases of the SCSI read and write commands. Programmed I/O mode is used during all other bus phases and SCSI commands.

The monitor is a straightforward implementation of the SCSI target functions. The monitor begins running on power-up or after pressing

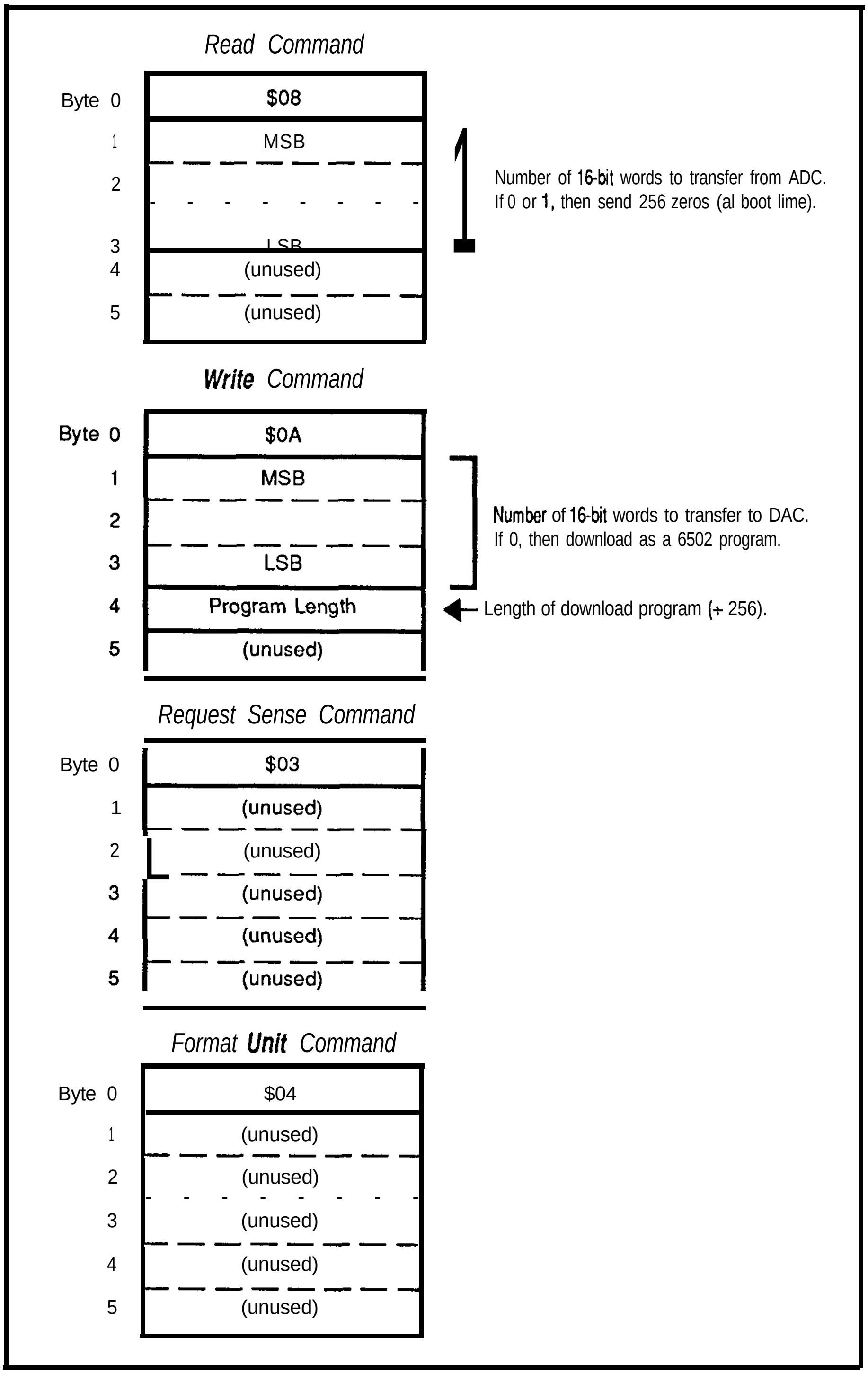


Figure 2—*DAQ3000* SCSI *command descriptor blocks*

the unit's reset button. The address of the monitor program ($3800) is stored three times in the EEPROM's last six bytes, where the 6502 microprocessor expects to find the addresses for interrupt, reset, and break routines. After initializing the system I/O (68B21) and the SCSI (5380) chips, the monitor polls the SCSI controller for detection of a SCSI bus selection phase directed toward the DAQ3000. Upon detecting a valid selection phase, the monitor asserts the command phase and begins to read the command descriptor block. The first byte of the command descriptor block determines which of the monitor's command handling routines is called. The DAQ3000 monitor contains subroutine handlers for performing the read, write, and request sense commands. (The format unit command is considered an error.) The monitor also contains utility routines for reading and writing SCSI data blocks using the programmed I/O and pseudo-DMA modes. For debugging purposes during SCSI command processing, the DAQ3000 displays simple status codes on the unit's eight front-panel LEDs, roughly indicating what part of the monitor program is being executed.

SCSIMOVER-A MACINTOSH APPLICATION

The second DAQ3000 software component, SCSIMover, is a relatively simple Macintosh application program that fulfills the Macintosh's role as the SCSI initiator for the DAQ3000. *[Editor's Note: Software for this article is available for downloading from the Circuit Cellar BBS or on* CIRCUIT CELLAR **INK** *Software On Disk #10. For downloading and ordering information, see page 78.1* SCSIMover was developed in Lightspeed Pascal. For simplicity and for portability to other Macintosh Rascal compilers, SCSIMover uses standard Pascal TTY-style I/O as its user interface. The current skeleton version of `SCSIMover` supports four commands: (1) receive digitized data from the DAQ3000 into Macintosh RAM; (2) send digitized data from Macintosh RAM to the DAQ3000; (3) download 6502 machine code to the DAQ3000 from a Macintosh text file; and (4) reset the SCSI bus.

`SCSIMover` calls SCSI routines in the Toolbox to perform the selection, command, status, and message-in phases. For the command phase, `SCSIMover` creates command descriptor blocks according to Figure 2. The remaining phases, data-in and data-out, are handled by two external 68000 assembly language procedures, `SCSIRFast` and `SCSIWFast`, which comprise the third component of the DAQ3000 software. These external procedures are necessary because the high-speed data transfers during the DAQ3000's data-in and data-out phases require the uninterrupted attention of the Macintosh's 68000 microprocessor. The standard SCSI read and write routines in the Toolbox cannot be used here because these

Toolbox routines allow the Macintosh's many event-driven interrupts to interfere with the SCSI transfer.

When calling **SCSIRFast** and **SCSIWFast, SCSIMover** supplies three parameters. The first parameter is a pointer to a memory area with which to receive or send the data samples. The second parameter is the number of bytes that will be involved in the transfer, and is user-specified. The third parameter is a pointer to an integer word that is to contain 5380 chip status information upon return from SCSIRFast or SCSIWFast.

FAST SCSI TRANSFER ROUTINES

SCSIRFast and SCSIWFast were written with the Macintosh 68000 Development System (now marketed by Consulair). The 68000 Development System produces relocatable object code that can be converted into a library file by a utility program included with Lightspeed Pascal. The library file is linked into **SCSIMover** by the Lightspeed Pascal compiler/linker. **SCSIRFast** and **SCSIWFast** follow standard Macintosh Pascal calling conventions, which specify how Pascal programs communicate with external assembly language subroutines through the Macintosh system stack. According to this convention, subroutine parameters, return values, return addresses, and local variables are organized in a block of stack memory called a stack frame.

When called, **SCSIRFast** and **SCSIWFast** disable all vectored interrupts, including interrupts generated by the keyboard, mouse, floppy disk driver, sound driver, and serial ports. The three subroutine parameters supplied by **SCSIMover** are then retrieved from the stack frame, the appropriate SCSI bus phase (data-in or data-out) is asserted, and a simple byte transfer loop is executed. **SCSIRFast** and **SCSIWFast** access the Macintosh's 5380 SCSI controller chip through direct memory addressing, thus bypassing all Toolbox routines. Direct addressing of the Macintosh's 5380 controller is potentially hardware dependent because it assumes a specific base address for the 5380 registers. (Apple does not guarantee that future models of the Macintosh will incorporate the 5380 at the same base address.) Unfortunately, direct addressing cannot be avoided because the Macintosh operating system does not provide, as it does for other I/O chips, a global variable containing the current 5380 base address.

POTENTIAL ENHANCEMENTS

The DAQ3000 system has a maximum 12-bit sampling rate of 28 kHz. At this maximum sampling rate, a Macintosh with 1 Mbyte of RAM can hold approximately 14.5 seconds of unpacked digitized data. A 28-kHz sampling rate corresponds to one 12-bit sample transferred every 36 µs (compared to the 25-µs conversion time of the ADC). The DAQ3000 monitor can be made to sample faster with some clever programming and/or elimination of some 5380 statuschecks. Transfer timing problems are likely to surface, however, since the **SCSIRFast** and **SCSIWFast** transfer routines cannot be made much faster.

As mentioned in the first part of this article, the function of all three DAQ3000 software components is independent of the DAQ3000's digital resolution. For a digital resolution of up to 16 bits, little or no software modificationswouldberequiredsince the DAQ3000 software already handles all sample data as full W-bit values. However, four bits are currently being "wasted" since the DAQ3000 has only 12-bit analog converters, so it may be desirable to implement a simple packing scheme in SCSIMover when storing data samples from RAM into a disk file.

Although SCSIMover is a relatively primitive driver program, it provides a solid shell for further development of application-specific functions. Since the DAQ3000 was originally designed for audio signal sampling, planned functions of SCSI - Mover include start/stop point editing, digital filtering, graphical display, and Fourier analysis of the digitized samples.+

John Eng recently completed a research fellowship with the Howard Hughes Medical Institute and is currently a senior medical student at the University of Wisconsin. His interest in microcomputingbegan with thepurchaseofan Apple][in 1981.

Bibliography

American National Standards Institute. *Small Computer System Interface (SCSI).* Document ANSI X3.131-1986, American National Standards Institute, New York, 1986. [Definitions of the mandatory command descriptor blocks can be found in this document.]

Apple Computer, Inc. *Inside Macintosh Volumes I–IV.* Addison-Wesley, Reading, MA, 1986. [This is the definitive documentation for routines in the Macintosh *Toolbox.* These volumes also cover various Macintosh conventions, such as those for the user interface and the calling of external subroutines.]

Chernicoff, Stephen. *The Macintosh Revealed: Volume 1, Unlocking the Toolbox.* 2nd ed. Hayden Books, Hasbrouck Heights, NJ, 1987. [Required introductory reading for the Macintosh programmer planning to implement a Macintosh-style user interface.]

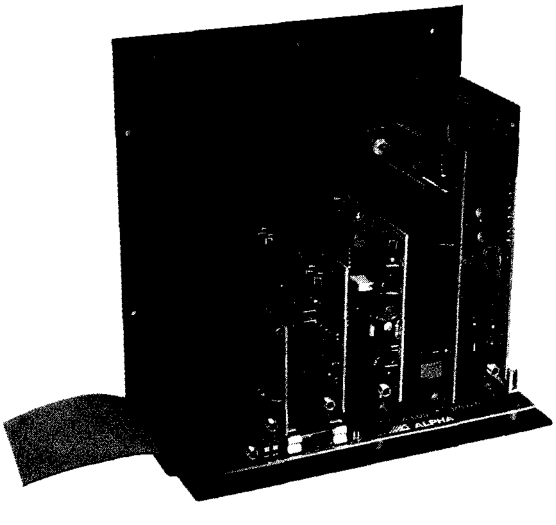
Chernicoff, Stephen. *The Macintosh Revealed: Volume 2, Programming with the Toolbox.* 2nd ed. Hayden Books, Hasbrouck Heights, NJ, 1987. [The companion to *Volume 1.*]

IRS

204 Very Useful
205 Moderately Useful
206 Not Useful

A-BUS™ MAGIC

Classroom to advanced industrial applications.



An A-BUS Motherboard with 5 cards.

NEW: REMOTE **A-BUS!** Use the new Serial (RS-232) Adapter or Processor to control any A-BUS system. Cards can be up to 500 ft away using phone type cable, or off premises using a modem. Call or send for the new A-BUS Catalog which covers all the products.

Be a Wizard in your Lab, Factory, College, Home...

It used to be difficult and costly to do process control, robotics, data acquisition, monitoring and sensing with your computer. Now the low-cost A-BUS system makes it easy to do almost any project you can imagine.

Versatility. A-BUS cards handle most interfacing, from on/off switching, to reading temperatures, to moving robot arms, to counting events, to sensing switches...

Adaptability. The A-BUS is modular, allowing expansion well beyond your needs. It works with almost any computer, or even as a remote data station with the new serial adapters.

Simplicity. You can start using the A-BUS in minutes. It's easy to connect, and software is a breeze to write in any language.

Reliabifii. Careful design and rugged construction make the A-BUS the first choice in speciafiied I/O.

An A-BUS system consists of: ▪ An A-BUS adapter plugged into your computer ▪ A cable to connect the adapter to 1 or 2 A-BUS function cards. ▪ The same cable will also fit an A-BUS Motherboard for expansion to up to 25 cards in any combination.

Important

All A-BUS Systems: ♦ Come assembled and tested ♦ Include detailed manuals with schematics and programming examples ♦ Can be used with almost any language (BASIC. Pascal. C. assembler. etc.) using simple 'IN' and "OUT" commands (PEEK and POKE on some computers) ♦ Can grow to 25 cards (in any combination) per adapter ♦ Provide jumper selectable addressing on each card ♦ Require a single low cost unregulated 12V power supply ♦ Are usually shipped from stock. (Overnight service is available.)

About Alpha Products

Founded in 1976 for the purpose of developing tow cost I/O devices for personal computers, Alpha has grown to serve over 70000 customers in over 60 countries. A-BUS users include many of the Fortune 500 (IBM, Hewlett-Packard, Tandy, Bell Labs. GM...) as well as most major universities. A-BUS products are U.S. designed, US. buff and serviced worldwide.
Overseas distributors: England: Caldy Science Assoc. Ltd., Merseyside. 051342 7033. Australia: Brumby Technologies Pty. Ltd., NSW, 759 1638. France: Cosem, Rungis, 46 86 64 75

Inputs, Outputs, etc.

Analog Input: 8 analog inputs. 0-5.1V in 20mV steps (8 bits). 0-100V range possible. 7500 conversions/second. **AD-142:** $142

12 **Bit** A to D: Analog to digital convener. Input range -4V to +4V, expandable to 100V. On-board amplifier. Resolution 1mV. Conversion time 130ms. 1 channel. (Expand to 8 channels with the RE-156 card.) **AN-146:** $153

Relay Card: 8 individually controffed industrial relays each with status LED's (3A at 120VAC contacts. SPST). **RE-140: $142**

Reed Relay Card: 8 reed relays (20mA at 60VDC, SPST). Individually controffed and latched. with status LEDs. **RE-156: $109**

D/A **converter:** 4 Channel 8 Bit D/A converter with output amplifiers and separate adjustable references. DA-1471 **$149**

24 line **TTL I/O:** Connect 24 input or output signals (TTL 0/5V levels or switches). Variety of modes. (Uses 8255A) **DG-148:** $72

Digital Input: 8 optically isolated inputs. Input can be 5 to 100V voltage levels or switch closures. **IN-141: $65**

Digital Output **Driver:** 8 outputs: 250mA at 12V. Drive relays, solenoids, stepper motors, lamps. etc. ST-143 $78

Clock **with** Alarm: Powerful clock/calendar. Battery backup. Timing to 1/100 sec. Alarm relay, LED and buzzer. **CL-144: $98**

Touch Tone Decoder: Each tone is convened into a number which is stored on the board. **PH-145: $87**

A-BUS Prototyplng card: 4x4.5' card. Will accept up to 10 I.C.s. With power & ground bus. **PR-152: $16**

Counter **Timer:** Three 16 bit counters/timers. Use seperately or cascade for bng (48 bit) counts. **CT-150:** $132

Call our application engineers to discuss your project.

Motion Control

Smart **Quad** Stepper Controller: **The world's finest.** On board microprocessor controls four motors simultaneously. Uses simple English commands like 'MOVE ARM 102 (INCHES) LEFT'. For each axis, you control coordinates (absolute or relative), ramping. speed, units, scale factors. etc. Many inputs for limit switches, etc. On the fly reporting of speed, position... Built in drivers for small motors (such as MO-193 or 105). **SC-149: $299**
Optiis: ▸ 5 amp/phase power booster for 1 motor: **PD-123: $49**
▸ Remote 'teach' keypad for direct motor control: RC-121: **$54**

SC-149 Smart Stepper **Controller.**

Stepper Driver Kft: For experimenting with stepper motors. Includes 2 MO-103 motors and a ST-143 dual driver PA-181: **$99**

Stepper Motors: (4 phase, unipolar)
MO-103: 2¼" dia. ¼" shaft, 7.5°/step, 12V. 5 oz-in torque. **$15**
MO-104: 2" dia. ¼" shaft, 1 .8°/step, 5V, 60 oz-in torque. **$45**
UO-105: 1.7' square. .2" shaft, 3.75°/step, 12V. 6 oz-in. **$15**

A-BUS Adapters

● Canaddrws64pontandcontrolupto25A-Buscards.
▸ Require one cable. Motherboard required for more than 2 cards.

A-BUS Parallel Adapters for:

IBM PC/XT/AT & compatibles. Uses one short or long slot.	**AR-133: $69**
Apple II,II+,IIe Plugs into any slot inside.	**AR-134: $52**
Commodore 64,128 Plugs into Expansion Port on back.	AA-13% $48
TRS-80 Model 102,200 Uses 40 pin "System bus".	**AR-136:** $76
Model 100 (Tandy portable) Plugs into socket on bottom.	**AR-135: $75**
TRS-80 Model 3,4,4D Y-Cable available if 50 pin bus is used.	**AR-132: $54**
TRS-80 Model I Plugs into 40 pin expansion bus.	AR-131: $39
Tandy Color Computers Fits ROM slot. Multipak or Y-Cable	**AR-138: $49**

A-BUS Cable: Necessary to connect any parallel adapter to one A-BUS card or to first motherboard. 50 pin, 3 ft. **CA-163: $24**
Special Cable for two A-GUS cards CA-162: $34

Serial Adapter: Connect A-BUS systems to any RS-232 port. Allows up to 500 ft from computer to A-BUS. **-SA-129: $149**

Serial Node: To connect additional SA-129/A-BUS systems to a single RS232 serial port (max 16 nodes). SN-128: $49

Serial Processor: same as above plus built in BASIC for off-line monitoring, togging. decision making, etc. **SP-127: $189**
Use SA-129 or SP-127 with modems for remote data acquisition.

Motherboard: Holds up to 5 A-BUS cards in sturdy aluminum frame with card guides. A sixth connector allows (using cables CA-161: $12) additional Motherboards to be added. **MB-120:** $108

Power Suppty: Power pack for up to 4 cards. **PS-126:** $12

Complete **Catalog** Avallable
For Orders and Info call (203) 656-I 606
Weekdays from 9 to 5 EST or FAX 203 656-0756

Ordering Information: We accept Visa, Mastercard, Checks. and M.O. C.O.D. is $4 extra. Purchase orders are subject to credit approval. CT residents add 7.5% safes tax.
Shipping: $4 per order (usually UPS ground). UPS 2nd Day Air: $4 extra Next Day service available. *Canada:* $6 par order (Airmail). *Outside US* and *Canada:* Add 10% of order total.

ALPHA Products
a Sigma Industries Company
242- C West Avenue, Darien, CT 06820

Circle No. 102 on Reader Service Card

FEATURE ARTICLE

Tracking Soviet Television Satellites

by Mark Dahmke

In recent years, due to lower cost and greater availability of satellite receiver technology, it has become possible to watch television programs from many other countries in this hemisphere. I've always had an interest in Europe and the Soviet Union and wanted to monitor their television, but due to the curvature of the Earth and the use of focused spot beams, there are limits to what can be seen from central North America.

In 1985 I learned that the Soviet Union operates a system of satellites in polar orbit which are used to broadcast television to northern Siberia. Because of their unusual orbits, they are visible to most of the northern hemisphere. I also discovered, to my surprise, that the technology to receive and track these satellites was developed in Omaha, Nebraska-a mere one-hour drive from my home in Lincoln. Fr. Lee Lubbers of Scola on the Creighton University pioneered theautomatictrackingsystemandhas installed many satellite systems for collegesand universities. With the help of Fr. Lubbers, Dan Pike, and Francis Lajba, I learned enough about the technology and methods of tracking Soviet satellites to build a receiver using a 2.4-meter dish in my back yard.

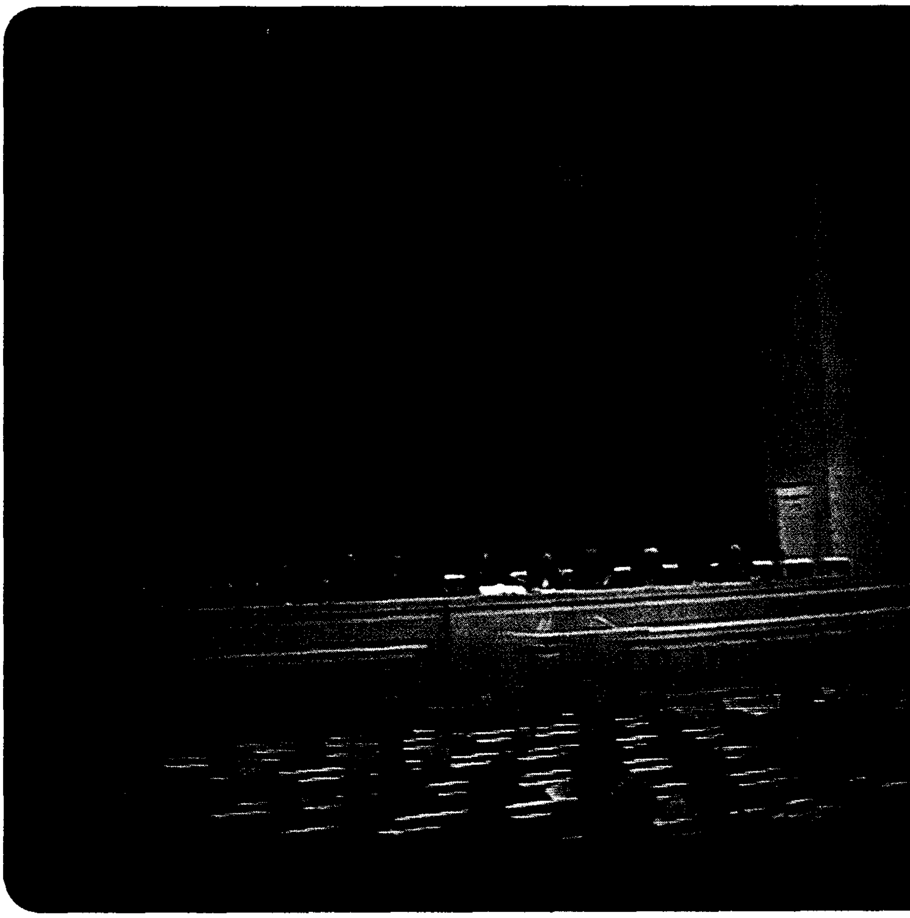


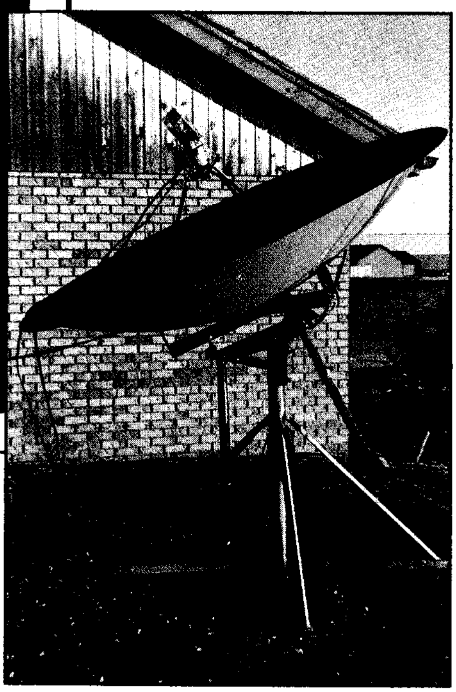
Photo 1 — *The author's eight-foot aluminum dish with modified polar mount. A standardpolar mount can be modified to provide dual-axis control of the dish. The elevation adjustment is controlled by a second actuator andjack so the dish may be moved from a 45-degree angle up to and past 90 degrees. The normal east-west motion is controlled by an actuatorjust as on a normal dish mount. Note that the pedestal leg on the left side of the mount in this picture points 20 degrees east of north.*

SIBERIAN LIGHTNING

In the mid-1960s, the Soviet Union needed a way to broadcast to the northern and eastern regions of Siberia. In the United States, we have been using geostationary "Clarke" orbits (named for Arthur C. Clarke, the inventor of the communications satellite) for such a long time that it's hard to think in terms of any other kind of orbit for broadcast television. The main advantage of a satellite in geostationary orbit is that you can point a dish at the satellite and never have to move it again. The trouble is that geostationary satellites aren't very useful for receiving sites above 70 degrees north latitude, and in the mid-60s the Soviet Union didn't have the capability to launch a large payload into a geostationary orbit anyway. The Soviets had other requirements, and have always been good at using available technology to meet their needs.

The Molniya orbit (pronounced Moln-ya, meaning “lightning” in Russian) has several unique properties. First, it is **a polar** orbit, inclined 63 to 65 degrees with respect to the equator. Two periods are possible: one just under 12 **hours and the other** just under 24 hours. The Soviets chose the **12**-hour orbit for their system. The **eccentricity is 0.74 which means** that it is highly elliptical. Figure 1 shows the properties of this orbit from a stationary point of view out in space. The orbit’s apogee (high point) is at about 40,000 km and the perigee (low point) is 600 km.

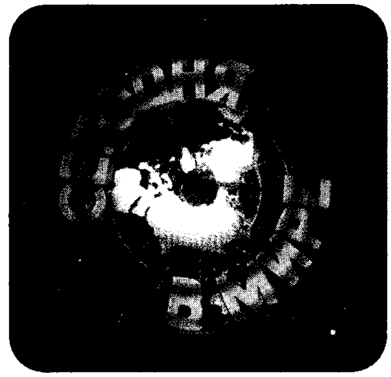
While remarkable in terms of conventional **geosta**tionary satellites, there are additional unusual features. Since the orbit has a period of 12 hours, the satellite will appear to be at its apogee over Hudson Bay, Canada during one orbit, then, because the Earth has revolved through

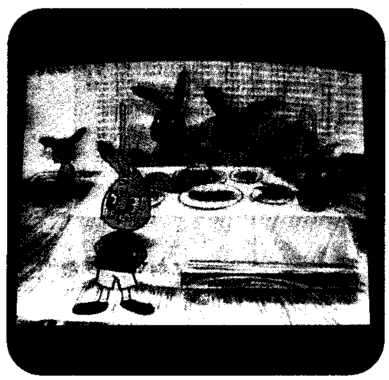


180 degrees underneath it, the next apogee will be over central Siberia. The effect of this combination of satellite motion and the Earth’s rotation means that the satellite will repeat its ground track very accurately each day. Only a limited number of stable ground tracks exist in this configuration, and the Soviets chose an ascending node of approximately 113 degrees West. Figure 2 shows the apparent ground track of this orbit.

From a location in the central United States, the satellite would appear to rise above the southern horizon, rapidly traverse the sky from almost due south to the zenith, then move into the northern sky. As it reached its apogee, it would appear to move more slowly and almost come to a stop, and generally never drop **below 60 degrees altitude in the north**ern sky. As it started to fall from its maximum altitude, it would move faster and begin to move back toward the zenith, eventually moving into the southern sky and dropping below the horizon. Figure 3 shows its apparent position in the northern sky. A satellite would repeat this track once per day because its next climb to apogee would be over Siberia and wouldn’t be visible to us in North America (actually, it would be visible but be very low on the northern horizon).

During the Canadian active loop, the satellite would appear to an observer in Siberia (or Moscow) to pop up from the northern horizon for a few hours, rise to about 30 or 40 degrees in elevation, then drop down and disappear until the next day. The “visibility window” for a given satellite may be up to eight hours. To solve this problem, the Soviet Union uses four satellites spaced six hours apart in nearly identical orbits, allowing for almost continuous coverage. When each satellite rises above the horizon, it is switched on and used for six hours, until control is handed off to the next satellite. The question most people ask is: why do they use the Canadian active loop, not the Siberian loop? There are two obvious answers: one is that both are used but since the **Sibe**rian loop is difficult to receive here, no one has really investigated this possibility. The second answer is that to track a satellite that is almost overhead means that the dish will collect a lot of snow and ice. A dish pointing at





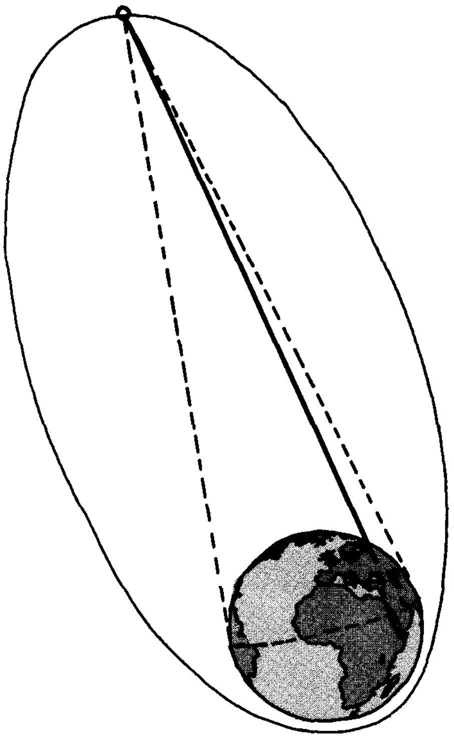


Figure 1 —The ellipse represents the 'true' 12-hour orbit, inclined 63.5 degrees out of the equatorial plane. The dashed lines show the signal coverage at apogee (maximum distance) which is approximately 24,860 miles above Hudson Bay, Canada. The solid line represents the signal uplink from Moscow. Each of four satellites is active for six hours each day, giving 24 hour coverage of Siberia,

Photo 2—The receiver and computer control setup. The monitor on the top shelf is a Sony PVM 127 113-inch multistandard unit. The receiver Is an older model Drake; the power supply to its righ t supplies 32 volts DC at about 1 amp, which powers the actuators. The PC is an XT clone with 10-megabyte hard disk and custom-built controller card with A-to-D input and relay controls.

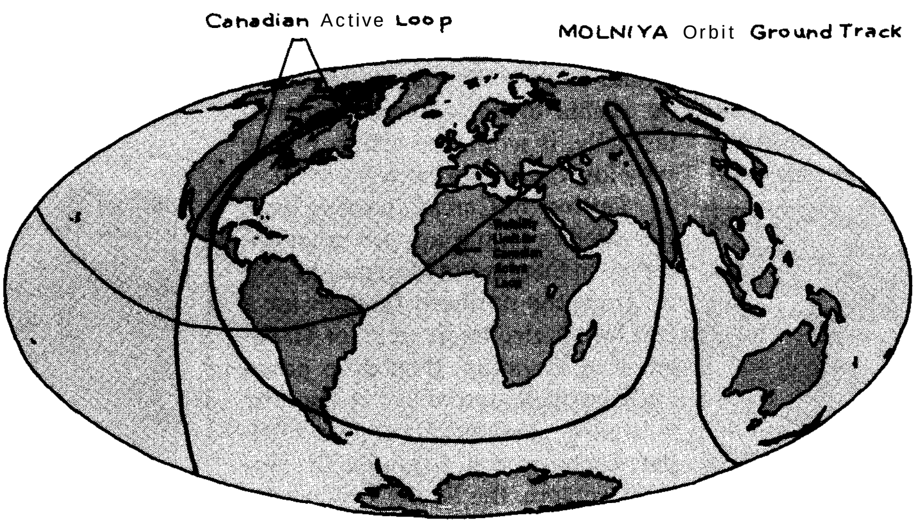


Figure 2—The dark line shows the apparent ground track of a typical Molniya orbit. Note the two active loops, one over Hudson Bay, the other over Siberia. The Canadian active loop is used for television broadcasts to Siberia.

a satellite above Canada would be placed at an angle comparable to that used to watch geostationary satellites, but looking north instead of south. Wind load would be a problem, but perhaps less a problem than ice load (it is widely believed that the standard receiving dish diameter used in remote receiving locations in Siberia is about 10 meters).

RECEIVING TECHNOLOGY

Since so much has been written lately about the differences between U.S. and European television technology, you might think that a design to receive Soviet television wouldrequire nothing but esoteric, hard-to-find parts. Surprisingly, it is possible to use off-the-shelf receivers, low-noise amplifiers, and feed horns with good results. The Molniya satellites broadcast in C-band just like domestic U.S. satellites, but use circular polarization rather than horizontal-vertical polarization. The single television signal available on the satellite has a downlink frequency of about 3874 MHz, or roughly half way between transponder frequencies 9 and 10 on domestic sa tellites. The FM audio subcarrier now used for TV audio is at 7.4 MHz. I've used a variety of low-cost receivers with excellent results, but for a first-rate signal and good control over bandwidth, a Chaparrel Sierra II or III receiver is ideal.

Since circular polariza tion is used, you must insert a small teflon dielectric block into the throat of the dish feed horn, which has the effect of rectifying or converting circular polarization into horizontal-vertical polarization. Without the teflon, you will experience a signal loss of 2 db, but with it, you will gain 2 db. The block can be ordered precut from Chaparrel Communications and other manufacturers.

The second major difference is that the Soviet Union uses the SECAM television standard (a French acronym for Sequential Color with Memory). *[Editor's Note: For more details on TV standards see Ed Nisley's article "ImageWise/PC" in* CIRCUIT CELLAR INK *#6.]* For the low-budget approach, a normal NTSC television will display a SECAM signal, but only in black and white and with the wrong aspect ratio. If you're serious about international television, you canbuy a multistandard receiver andVCRfrom companies like Sony and Hitachi.

Prior to September 1988, the Soviets used a modified SECAM format in

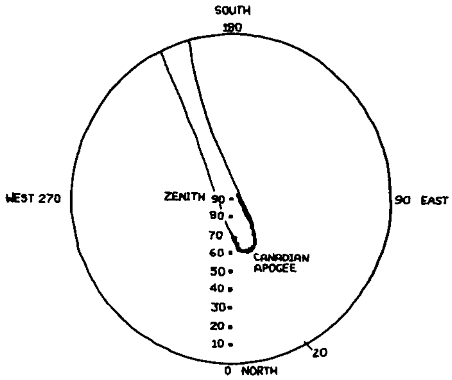


Figure 3—*The Molniya orbit as seen from the miciwest. The dark part of the curve shows the active portion of the orbit-the period used to broadcast television.*

which the TV audio was added as a pulse next to the horizontal sync in each scan line. This "sound in sync" method wasadopted before there were any standards for the transmission of audio on an FM subcarrier. The old methodusedpulse-widthmodulation and required a decoder circuit that would extract the audio using sample-hold techniques and then reconstruct a normal SECAM sync pulse for the monitor. Fortunately this technique has been phased out, greatly simplifying reception of Molniya broadcasts.

The signal strength from these satellites is quite high, comparable to the powerful transponders of Galaxy I. I've received marginal quality pictures using a mere six-foot dish and 135-degree LNA (Low-Noise Amplifier). My standard setup, shown in Photo 1, is an eight-foot (2.4-meter) solid aluminum dish with a 60-degree LNA (the noise temperature in Kelvin of an LNA rates its ability to amplify signals above the noise in the amplifieritself, thebackground radiationof the Earth, and from outer space). A few years ago, good-quality, low-temperature LNAs were extremely expensive, so the only option was to move to a larger diameter dish. Now, a 60-degree LNA costs less than a hundred dollars.

TRACKING A MOVING SATELLITE

Now comes the fun part. It is possible to manually locate and track the Molniya satellites, and I've done so many times. But to really make this a fun project, and to set up an automated receiving station that a non-techie can use, we need some computer-controlled motors and analog-to-digital converters.

When planning to build a Molniya receiver, your location should be taken into account. In the central United States an eight-foot dish will work well, but if you live farther south, a ten- or twelve-foot dish would be advisable. A sixteen-foot dish maybe required at or near the visibility limit shown in Figure 2.

DISH POSITIONING

The dish you use must be steerable with two degrees of freedom. Several axis arrangements are possible: A simple azimuth-elevation mount will work, with rotation (azimuth) and elevation (altitude). The disadvantage of this approach is that the satellite often must be followed across the zenith point, which also happens to be on the rotational axis of the dish mount, resulting in a singularity. That is, you would be pointing straight up with the ability to rotate the dish around the azimuth axis, but it would always point to the same spot. A better arrangement is to have the rotational axis offset to someother part of the sky. Such an offset is easily accomplished by taking a standard polar mount (where the axis of rotation points to the north celestial pole) and swing it around 180 degrees, with the polar axis pointing due south. A standard dish mount with an actuator (a DC motor with worm drive) can then position the dish to the east and west, while an additional actuator is used to raise and lower the dish in altitude (see Photo 3). Any number of dish configurations are possible depending on the available hardware and dish mounts. The only requirement is that the dish be able to move freely through about 45 degrees of altitude and 45 degrees of azimuth. The actual center point of this "window of visibility" is dependent on your location with respect to Hudson Bay, Canada.

remember exactly what position the arm is in, or recalibrate each time. The circuit in Figure 4 shows the complete controller. This design is used with a low-cost PC/XT clone motherboard, a CGA display, and one floppy disk drive, but provides all the necessary functions to track satellites.

If you don't know where the currently active satellite is, you will have to hunt for it manually. This can be done by hand with momentary switches connected to the controlling relays or by computer control in a scanning pattern. The position of an active satellite can only be determined by looking at RF signal strength, and by watching for a TV picture or listening for the audio at 7.4-MHz.

My house happens to be directly in the path of a strong terrestrial microwave signal which causes false triggering and often shows a high RF signal strength when in fact no satellite signal is present. For this reason, a fully automated tracking program doesn't work well. If you live in an area that is free of microwave interference, you will be able to track each satellite and automatically hand off to the next one with little difficulty. In spite of the microwave interference, I don't have much trouble with automated tracking once I manually locate the satellite and lock onto it, but if the position of the satellite happens to be next to a strong terrestrial source, the tracking program can become confused and be dragged off into the microwave noise. Also, due to these hot spots in the sky, I can't run an automated X-Y scanning procedure to locate the active satellite. At SCOLA's site in Omaha, and at Lincoln High School where I have helped to install a system, the microwave interference is not a problem and the automated tracking and scanning software work as they should.

At first, finding the satellite can be very frustrating, but some simple guidelines will help. First, check a map to find the bearing from your location to the north end of Hudson Bay, Canada. Specifically, the apogee

Photo J-Close-up view of the modified polar mount.

is always near 80 degrees West Longitude and 63 degrees North Latitude. Position the dish to the midpoint of its east-west actuator range. Find the offset from true north to the apogee location and aim the dish at that point. For the continental U.S., set the midpoint of the dish elevation range to be your latitude plus 30 degrees. In Europe, set the midpoint to be about the same as your latitude. Manually pan the dish, scanning through the entire "window" in four-degree increments or bands. With the receiver set to transponder 9 and audio subcarrier at 7.4 MHz, you should encounter at least a distorted picture and some sound. After finding maximum signal strength, you can adjust the receiving frequency and other parameters on the receiver for a clear picture.

One of the best ways to learn how the satellites move is to manually track them through one or more active loops. If the satellite is near apogee, you won't have to readjust the dish position more than once every fifteen minutes or so, but if the satellite is starting back down toward perigee (moving higher in the sky), it will be accelerating and will require adjustment at least every four or five minutes. After you have determined the optimum range of pointing angles for your dish, you can adjust the actuator mounting brackets to allow for the optimum range of motion. Remember that each satellite will be in a slightly different orbit, so they won't always appear in the same place in the sky as another satellite.

One other complication: due to the parameters of the Molniya orbit, these satellites recede in their orbits by about 4.5 minutes per day. This means that if you found the satellite at a certain point today, you would find it at that exact same point 4.5 minutes earlier tomorrow. Assuming that you are tracking the satellites automatically each day and every day, the practical effect is that the hand-off times (when control is transferred to the next upcoming satellite) will be four and a half minutes earlier every day. Since this is a small change on a daily basis, the software will take it in stride, but if you happen to skip a few days or a week or two, you'll probably have to go satellite hunting again (unless you have a really good memory). I use my system a couple of times a week, and with a bit of practice, I have found that it takes me less than two minutes to find the currently active satellite, and in the worst case, perhaps five minutes.

If you follow Molniya satellites for months or years, you will also discover that sometimes their orbits

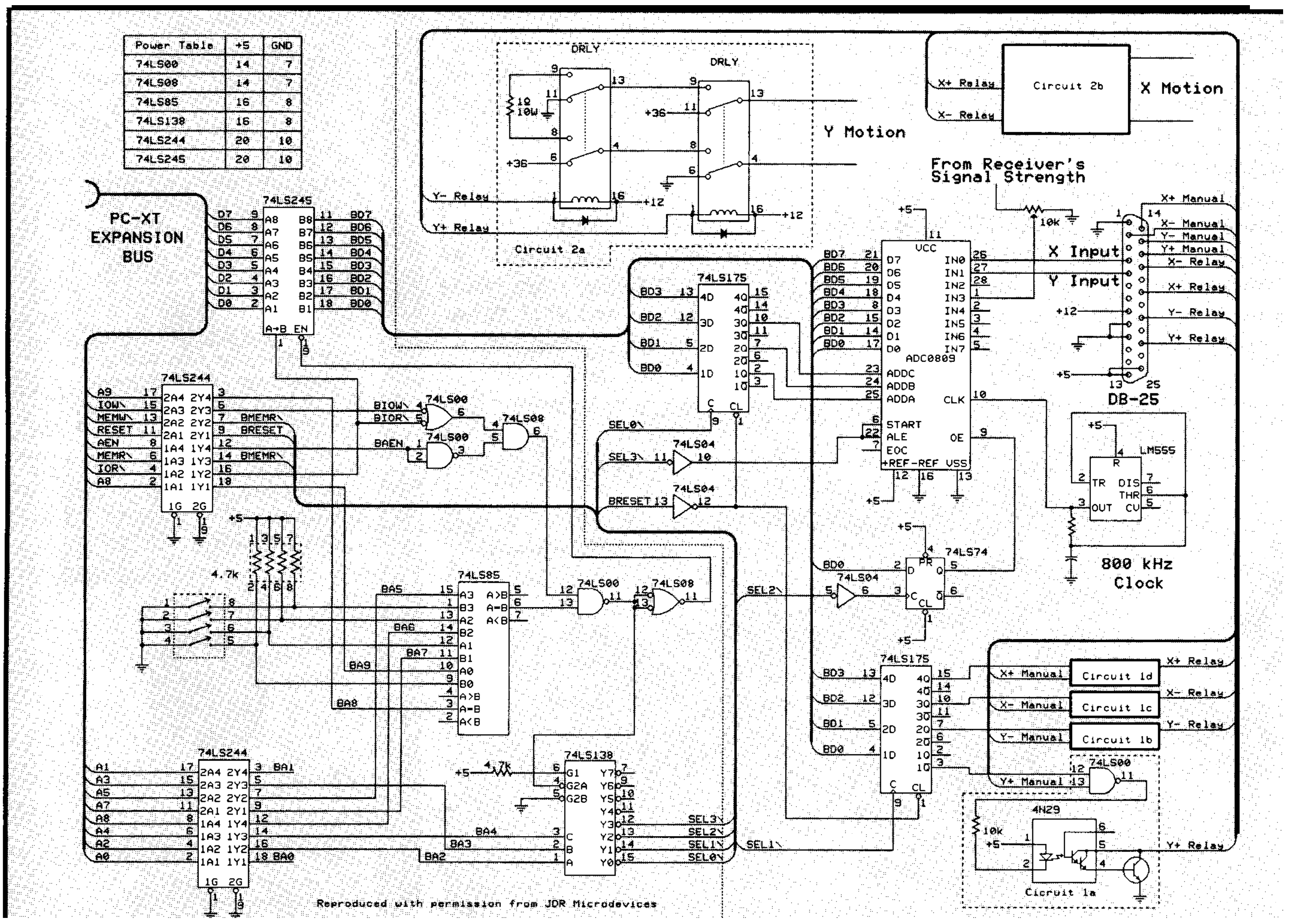


Figure 4—The dish controller circuit for use with a PC-compatible. The ADC0809 8-channel analog-to-digital chip was used to read both X- and Y-axis voltages which provide feedback about dish position. and to read the RF signal strength which is fed from the satellite receiver. Four output lines feed relay controls which are combined to produce X and Y actuator motor voltages. The DC motors are reversible depending on polarity of the 32-volt input.

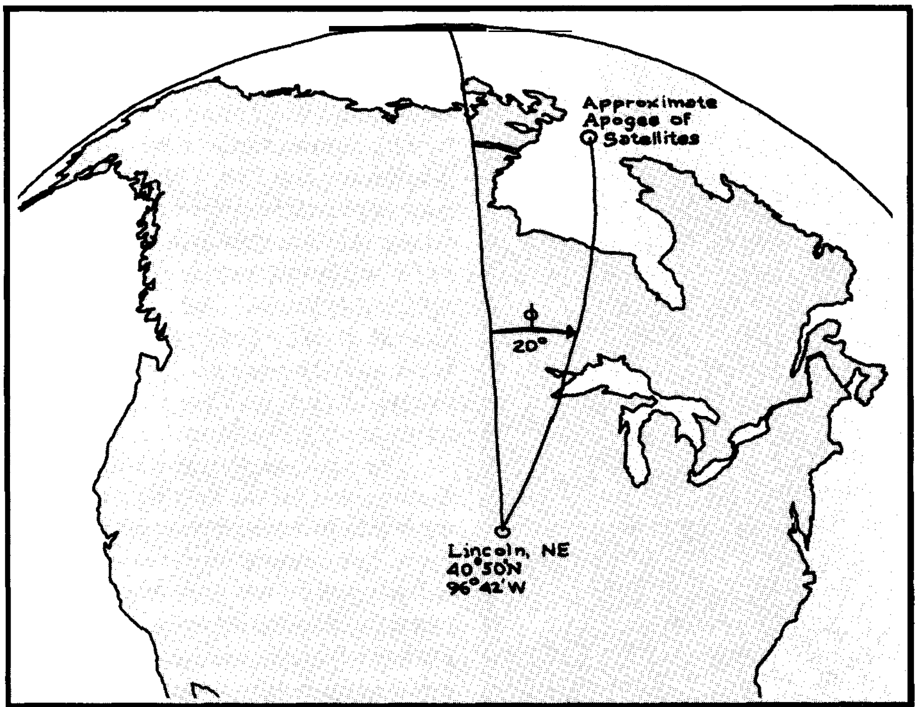


Figure 5—*Determining the angle from the receiving antenna to Hudson Bay, Canada. Depending on your location, the dish should be aligned so that the center of its range of movement is offset from true north. For Lincoln, Nebraska this should be about 20 degrees east of north. For the east coast, the dish should point due north. For southern California use a 30-degree offset and for northern California use a 40-degree offset.*

```
step2 = 1.0                       'course position increment
step1 = 0.5                       'fine position increment
for step = step2, step1           'two loops: step = 1.0,
                                  '           step = 0.5

loop1 : signal = oldsignal        'update signal
        move-up(step)             'move one step
        signal = getRF()          'get new signal
        if signal > oldsignal then goto loop1
                                  'if greater, then
                                  'do it again
        move-down(step)           'else return to
                                  'starting point

loop2:  signal = oldsignal        'repeat for
        move-down(step)           '-Y direction
        signal = getRF()
        if signal > oldsignal then goto loop2
        move-up(step)

loop3:  signal = oldsignal         'repeat for
        move-left(step)            '-X direction
        signal = getRF()
        if signal > oldsignal then goto loop3
        move-right[step)

loop4:  signal = oldsignal         'repeat for
        move-right(step)           '+X direction
        signal = getRF()
        if signal > oldsignal then goto loop4
        move-left(step)

        next (step)                'repeat for
                                   'fine positioning
        end
```

Listing 1—*The search algorithm moves the dish in four directions looking for the maximum signal strength. The first pass is designed for most rapid movement, while the second is optimized for fine tuning.*

```
FUNCTION GetRF()
CONST B = 3                 'ADC0809 channel address for RF sig
CONST SEL0 = &h300          'Latch ADC address
CONST SEL1 = &h304          'Latch relays
CONST SEL2 = &h308          'Output enable ADC data lines
CONST SEL3 = &h30C          'Start ADC conversion
                            'get signal strength
FOR k = 1 TO 4              'read 4 times
  OUT SEL0, b               'set up the ADC0809 channel addr
  CALL WAIT(l)              'wait a millisecond
  OUT SEL3, 0               'start ADC conversion
  CALL WAIT(l)

  OUT SEL2, 1               'Output enable ADC data lines
  CALL WAIT(1)
  siga = (INP(a + 20)) + siga ' accumulate
  OUT a + 8, 0
  NEXT k
  GetRF= siga / 4            'average four readings
  RETURN

  SUB GetXY(xc,yc)

' get x,y coordinates

yc = 0
xc = 0
FOR k = 1 TO 2
  b - l                      'data channel 1
  OUT SEL0, b                'load addr
  CALL WAIT(l)
  OUT SEL3, 0                'start conv
  CALL WAIT(l)
  OUT SEL2, 1
  CALL WAIT(l)
  xc = XC + INP(a + 20) 'read the result
NEXT k
xc = xc / 2                  'average

FOR k = 1 TO 2
  b - 0                      'data channel 0
  OUT SEL0, b                'load addr
  CALL WAIT(l)
  OUT SEL3, 0                'start conv
  CALL WAIT(l)
  OUT SEL2, 1
  CALL WAIT(l)
  yc = yc + INP(a + 20) 'read the result
NEXT k
yc = yc / 2                  'average
RETURN
```

listing 2—*This QuickBASIC code determines signal strength by averaging four readings and then comparing averages.*

change unpredictably, due to the use of station-keeping thrusters. Just like geostationary satellites, atmospheric drag and gravi tational forces will eventually alter an orbit and require some intervention. Also, whenever a satellite grows weak or fails, a new one must be launched, often with different orbital parameters. In January of 1988 one of the four satellites apparently failed. Normally a replacement is launched immediately, but for some reason this didn't happen until July. No direct statement could beobtained from the Soviet government, but some thought this meant they were discontinuing the use of the Molniya satellites; others had heard rumors that the replacement satellite and launch vehicle blew up on the launch pad and that no spare was available, causing the delay. In July the gap was filled, which was a great relief to those of us whoenjoytuninginto"theotherside."

THE SOFTWARE

The primary function of a simple Molniya tracking program is to maintain maximum signal strength by repositioning the dish. Other functions such as remembering from day to day where each satellite is or implementing a scanning search program are optional features. Once a satellite is located (under program control or manually) the program should be able to keep track of it for the remainder of its six-hour active loop.

Through experimentation, I've discovered that the best way to track these satellites is to reposition the dish every two minutes regardless of signal strength. My first version updated the position if the signal strength fell more than 20 percent since the last position update, but due to terrestrial microwaveinterference, this technique was unreliable. Again, if you live in an area that is free of such background interference, this approach will work quite well.

Every two minutes, my program moves thedish through a programmed search pattern-up, down, left, and right of the current position. The amount of movement depends on many things including the size of your dish and signal strength. The goal is to move the dish only enough to detect a change in signal strength, but not enough to fill the picture with static. The viewer should not be aware that the dish is moving. This type of adjustment involves changing the duration of relay closure. The search algorithm I use is shown in Listing 1. *[Editor's Note: Software for this article is available for downloadingfrom the Circuit Cellar BBS and on Soffware On Disk #10. For downloading and ordering information, see page 78.1*

This loop will move the dish in all four directions looking for maximum signal strength. The first pass is with a step size of 1.0—meaning the full relay closure time (maybe 300 ms), while the second pass sets the step size to half of this value for fine tuning.

The dish movement routines `move-up()`, `move-down()`, `move-left()`, and `move-right()` must close the appropriate relay for a specific number of milliseconds. As mentioned before, the time delay used here must be determined experimentally depending on the type of actuator used. The GetRF () routine simply reads the analog-to-digital converter and returns an 8-bit value for RF signal strength.

Dish position can be determined by reading the A-to-D converter channels 0 and 1. Positions can be stored each day and can be used to represent the X-Y position on a graphics display, making manual location of satellites easier if they are tracked on a regular basis. I've also added a `GotoXY(x,y)` routine to my package to quickly reposition the dish. The QuickBASIC code for GetRF () is shown in Listing 2.

EAVESDROPPING ON THE SOVIET BLOC

Access to Soviet television programming which is intended only for a domestic audience provides a rare opportunity to see the Soviet Union without the filtering that exists when material is selectively chosen for an American audience. It provides a view of the country and its people as they see themselves. Few people realize that the Soviets have their own version of TV ratings and market research, causing fierce competition among producers and writers to respond to the demands of viewers. Not surprisingly, the variety and quality of programs has increased during the Gorbachev era. The photos at the beginning of the article show some examples of Soviet TV programs.

Although there are many news and current affairs programs, the national networks provide a rich variety of cultural programs from virtually every Soviet republic, along with sports, travelogues, game shows, exercise programs, documentaries, spy thrillers, World War II movies, concerts, children's cartoons.. .✣

Mark Dahmke is a consulting editor *for* CIRCUIT CELLAR INK. *He lives in Lincoln, Nebraska and works as a consultant for the Lincoln Telephone Co. and professional research consultant in Omaha.*

Complete Molniya tracking systems are available from:

SCOLA
(Satellite Communications for Learning)
California at 24th Street
Omaha, NE 68178
(402) 280-4063

SCOLA **also operates a** subscription service to live or near-live international television via Telstar 303, including daily programming from a dozen countries including The USSR, China, Japan, Italy, Germany, Mexico, Iran, and France.

Teflon dielectric blocks for circular polarization conversion are available from:

Chaparrel Communications
2360 Bering Drive
San Jose, CA 95131
(408) 262-2536

Further reading:

The Satellite TV Handbook
Anthony T. Easton
Howard W. Sams & Co., Inc.
ISBN 0-672-22055-5

IRS

207 Very Useful
208 Moderately Useful
209 Not Useful

Circle No. 138 on Reader Service Card

Circle No. 120 on Reader Service Card

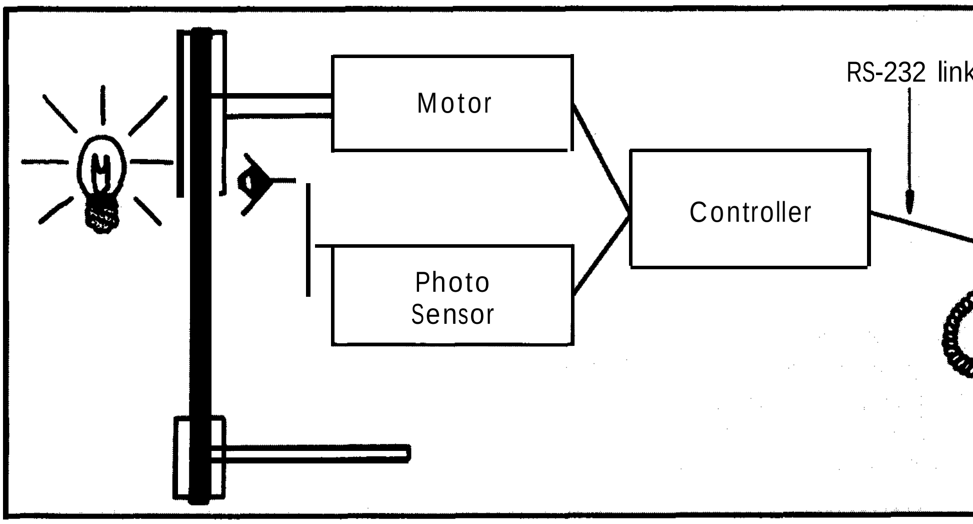
FEATURE ARTICLE PART I

A Network for Distributed Control

Building an RS-485 Network for Controllers

by Ed Nisley

Around the Circuit Cellar, hardware designs frequently occur after someone throws down the gauntlet.



That someone is often Steve, who needs "just a few things" for his next project. The recent RTC boards are an example: myriad functions in a form factor you can't pass up at a price you can't ignore. He needed something like that for his No Sparrow Shall Fall home security system, of course.

But, having gotten Jeff started on the RTC hardware, Steve slapped down the other glove. He figured there was no reason why the RTC boards couldn't talk to each other; what he really wanted was a network of controllers to distribute data collection and analysis functions throughout his system. Talk of expensive communication controllers was verboten, as Jeff had already included an RS-485 transceiver on the board. The rest was just software.. .

This two-part article describes the result: INKnet, an RS-485 network designed for microcontrollers. The firstpartcoversnetworkperformance and access protocols, while the second part will explore INKnet messages and the firmware required to move them around the network.

THE OLD STANDBY

Most controllers include an RS-232 port, probably set up as shown in Figure 1. The controller connects, through a "standard" RS-232 cable, to any terminal or PC. The data rate can exceed 100,000 bits/second and the cable can grow beyond 50 feet, as long as you don't stretchboth limits simultaneously. Best of all, everyone understands how to program RS-232 communication routines!

However, that ubiquitous RS-232 standard defines just the connection between Data Terminal Equipment (a terminal or computer) and Data Communication Equipment (a modem). The RS-232 electrical spec implicitly assumes the cable connects one driver to one receiver. Although you can wireseveral receivers toa singledriver, there is no practical way to use several drivers with a single (or multiple) receiver without ugly and nonstandard circuitry.

RS-232 requires a separate serial port for each connection, so systems with even a few controllers begin to sprout a rat's nest of cables. If each controller communicates with the outside world the situation is even worse, because the setup begins to look like the bridge of the Enterprise. Figure 2 shows a system with three controllers; obviously unworkable unless you think a deskcrowded with PCs is a status symbol.

A separate controller can handle all serial communication with the outside world, but this doesn't reduce the confusion of ports and software in the system. Worse, the communications controller must somehow mesh all of the traffic with the PC's screen and keyboard. You can imagine how difficult it is to expand this type of system with another controller.

RS-232 was obviously never intended as a network standard. There must be better ways to link multiple controllers than by bending RS-232 around a problem it wasn't designed to solve.

A MATTER OF SIMPLE HARDWARE

The RS-485 communication standard improves on RS-232 without throwing the baby out with the bath water. Up to 32 transceivers (a transmitter and receiver pair) can be connected by a single pair of wires. The interface circuitry uses a single five-volt power supply. Best of all, the same TTL-compatible serial ports that drive RS-232 circuitry will handle an RS-485 transceiver.

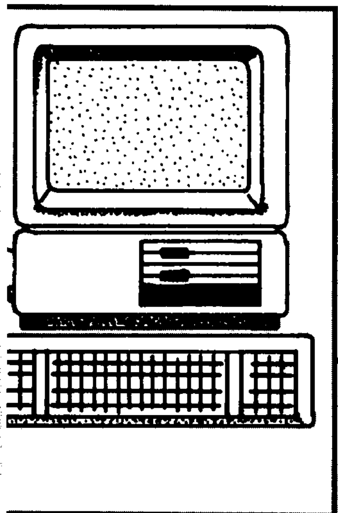


Figure 1—*Typical RS-232 connections are between a computer and a single device.*

Figure 3 shows an RS-485 network version of Figure 2. The wiring between controllers is ordinary twisted pair. The PC may use either an RS-485 card or an adapter on a standard RS-232 card. Each controller or PC is called a "node" to indicate that it connects at a single point and counts as one of the 32 allowed transceivers.

There are some limitations to an RS-485 network. The twisted pair should have a 100-ohm terminating resistor at each end, thus the nodes must either be daisy-chained or connected to the main wire with short unterminated stubs. The maximum wire length is about 1300 meters overall, and the two conductors are not interchangeable, so you must wire them the same way to each transceiver.

Although up to 32 transceivers can share a single twisted-pair connection, only one driver may be active at a time. The RS-485 standard does not specify how this is to be achieved, but, fortunately, the hardware can handle brief shorts and misconnections. Two nodes trying to transmit at the same time will combine to produce easily detected mush at the receivers without damaging the transmitters.

RS-485 drivers produce a differential voltage on the twisted-pair wires. In this context, differential means that the signal is measured between the two wires rather than from either conductor to ground. The advantage of a differential signal is that, while noise pulses will be coupled equally onto both wires, the receiver

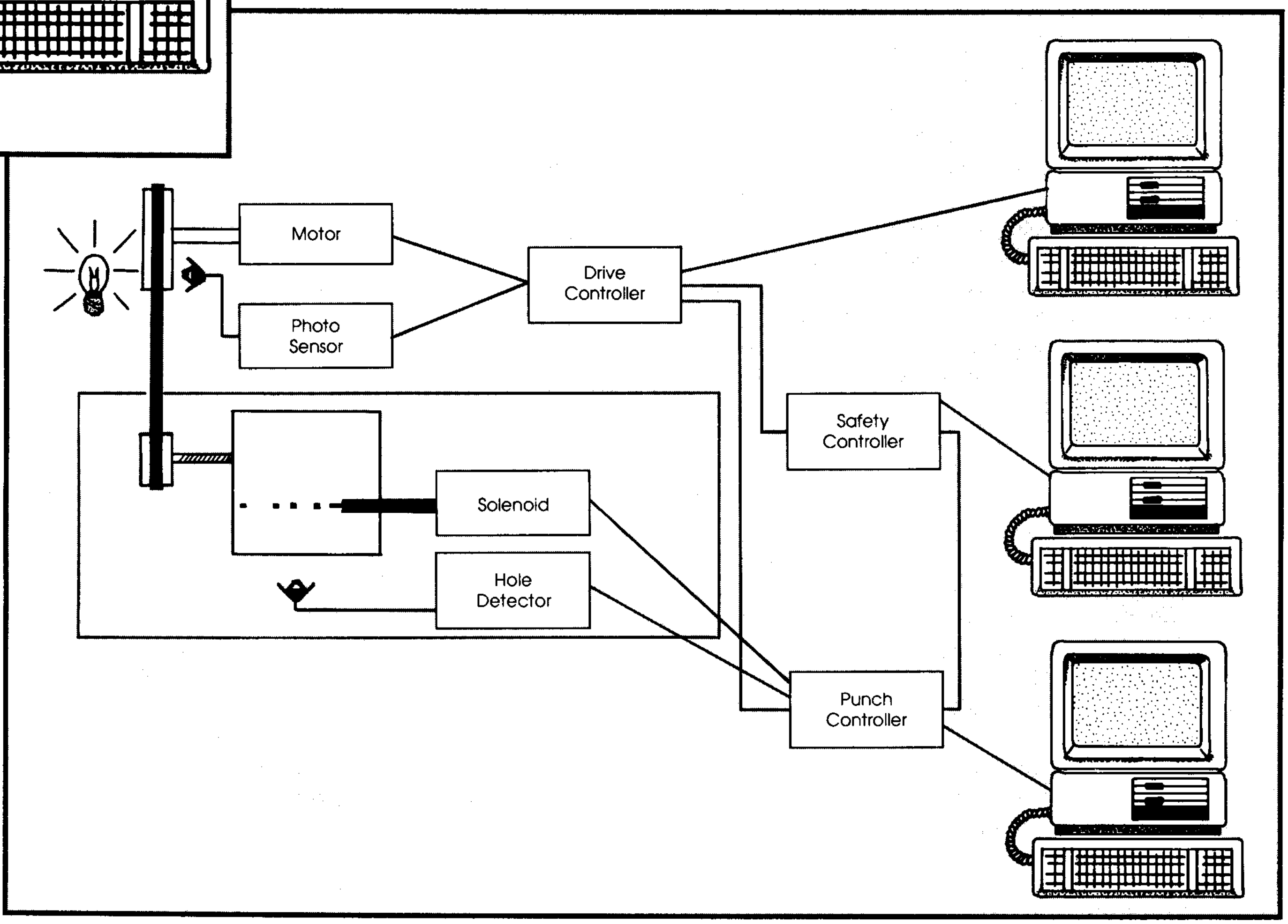


Figure 2—*When multiple computers need to communicate with multiple devices, use of RS-232 can result in a rat's nest of connections.*

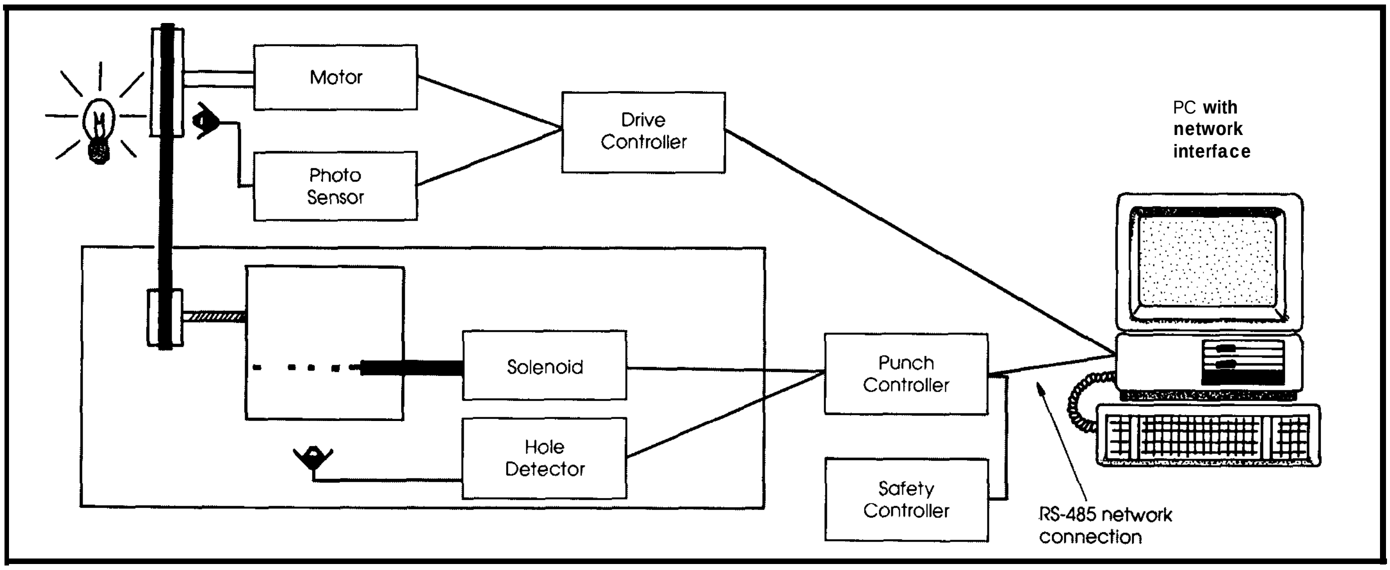


Figure 3—*Use or RS-232 reduces multiple connections to a single twisted-pair wire.*

will ignore any signal that appears on both its inputs.

You can use any pair of wires, but the best noise immunity is gained when the wires are twisted a few times per foot. Zip cord will work for desktop connections, but don't try running it down the hall! For particularly long runs you should match the characteristic impedance of the cable with the terminating resistors.

Because the twisted pair establishes a metallic connection between all nodes, it may carry ground current between nodes in addition to the desired signal. If a severe ground voltage imbalance exists between two nodes, this current can destroy the interface circuitry. All nodes must be either isolated from ground or bonded to the same ground point to eliminate any danger from potential ground loops.

Perhaps this goes without saying, but all nodes must use the same bit rate, number of data bits, parity bits, and stop bits. RS485 offers no magic resolution to this familiar RS-232 problem.

The RTC52 schematic in "From The Bench" in CIRCUIT CELLAR INK #8 shows that a complete RS485 interface can be a single 75176 IC with a jumper for the 100-ohm termination resistor. The 8052 (or 8031) handles data input and output through the standard on-chip serial port.

BASIC BANDWIDTH

Networking does not eliminate all communication problems in a single stroke. As with all engineering decisions, choosing to network your embedded controllers is a tradeoff between two sets of problems and op portunities. Make sure your application is amenable to a networked solution because there is nothing to be gained by force-fitting the wrong answer to the right problem!

Evaluating a network requires thinking about your control programs a little differently. Instead of looking at raw computer power, you must evaluate how to divide the control algorithm between multiple processors, what information flows between the processors, and whether there are any timing requirements for that data. All of these issues boil down to making sure that the network will deliver enough information in a timely manner.

More often than not, the limiting factor will be the speed of the network-no matter how fast it is, you will always need more speed! Unfortunately, determining whether a network can handle your project is not simple. You must take into account not only the network's capacity, but how your program will use the network; everything is related to everything else.

The fundamental limitation of any communication system is the total amount of data it can handle in a given time, which is usually measured in bytes/second or messages/second. Those of you familiar with information theory will recall Claude Shannon's seminal work defining information transfer through noisy channels. The bottom line is simple: regardless of how bright you are, you cannot transmit more information than the system can handle!

There are clever encoding and compression techniques that give the appearance of stuffing more data through a wire than seems physically possible. These tricks simply make better use of the available bandwidth. The "bandwidth" of a communication system is simply the highest data transfer rate for the particular encoding scheme being used.

For our purposes, the network bandwidth is the maximum number of bits per second appearing on the wires. In the case of an RS485 or RS-232 connection this is the transmitter data rate. For example, if the nodes transmit at 5000 bits/second, the network can handle 5000 bits every second. This conclusion may not sound earthshaking, but if your application must exchange 1 million bps there is an obvious data rate mismatch and the network won't solve your problem.

I will use 5000 bps in these examples to make the math easy to follow. You can apply the calculations to find the capacity of any network, regardless of its data rate. Remember that the data rate depends on the network hardware, the length of the interconnections, the speed of the firmware, and myriad other factors. The actual data rates are generally much higher than 5000 bps.

Because only one RS485 transmitter can be active at a time, the available bandwidth must be divided among the nodes. Suppose that there are five nodes; dividing the bandwidth "fairly" gives each node 1000 bits per second. Ten nodes each get 500 bps, and so on.

The bandwidth need not be divided equally between all the nodes, but the total network bandwidth cannot be exceeded. If one node transmits 4000 bps, the remaining nodes must divide the remaining 1000 bps among themselves. If one node hogs the network by transmitting continuously, no other node can say anything!

Because serial ports transmit complete bytes of data instead of individual bits, a more useful measure of network bandwidth is bytes per second. The familiar asynchronous communication format of one start bit, eight data bits, one stop bit, and no parity bits requires ten bit times to transmit eight data bits, so a 5000-bps network can transmit 500 bytes per second.

The alert reader winced at that simple division, because 20% of the potential network bandwidth has quietly vanished into the start and stop bits around each byte! This also happens in ordinary asynchronous RS-232 communications, but, because we rarely analyze the situation so carefully, it usually goes unnoticed and unremarked.

Why Twisted Pair?

Sending an electrical signal between two locations requires a pair of conductors separated by an insulator. Choosing the right conductors, insulator, and physical arrangement depends on both the signal and the environment.

Some signals can use nearly any conductors. For example, doorbell wiring is completely non-critical as long as the wires can take the mechanical abuse. 1 have seen a doorbell circuit with only one wire; the other conductor was the aluminum house siding!

The two wires carrying the signal also serve as an antenna to transmit or receive electromagnetic interference. Recall that a changing current will create a changing magnetic field, while a changing field will induce a current. In most electronic applications the induced current is undesirable noise.

The current induced in a pair of wires is proportional to the strength of the magnetic field and the area between the wires through which the field passes. Because the strength of the field is not under our control, the only way to reduce the current is to reduce the area. This can be carried only so far, however, because the wires must be separated by some insulation. Figure A shows this effect.

However, the induced current depends on which way the field passes through the wires. Putting a "flip" in the middle of the wires, as shown in Figure B, causes the field to pass through equal areas in opposite directions. As a result, the total current induced is zero!

Because the direction of the magnetic field is not known (and is not uniform, anyway), twisting the wires together into a double helix is the best compromise. The currents induced in each half-turn will largely cancel those in the next half-turn regardless of the field direction.

Twisted pair wiring has other desirable features, such as balanced impedances, but for our purposes it is enough to say that it works well and is inexpensive.

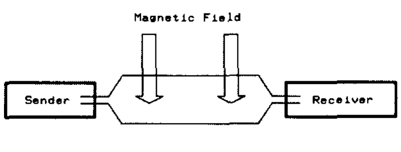


Figure A

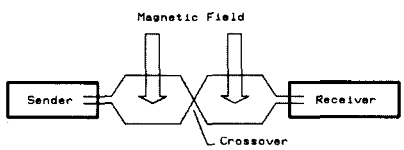
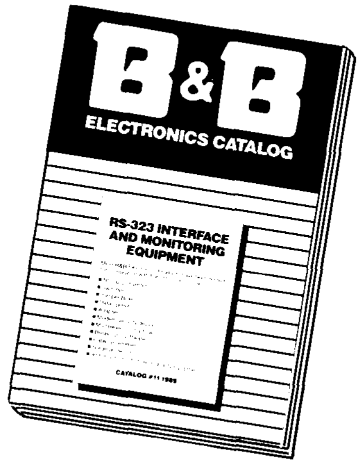


Figure B



PACKAGED DATA

An RS-232 link between two CPUs allows no question about who is sending data to whom: there is one sender and one receiver. With an RS-485 network, however, all receivers "hear" every message sent on the network. The messages must include some address information to specify which node should act on the data.

Each message should also include the address of the node sending the message, so the receiver can tell where the data originated. In some cases this will make no difference, as with an LCD panel that simply displays whatever text comes along, but some nodes must know who sent the message to make sense of the contents.

Each message must include some indication of when it's complete. This can be accomplished with a unique terminating character or a length count. A terminating character is useful only if that value cannot occur in the data, but most networks carry binary data which can include all possible byte values. Including the length of the **message** is unambiguous and requires very little overhead, so it is the most common method.

Because different nodes send different types of data, the message must include an indication of what the rest of the data "looks like." This code distinguishes BASIC source code statements, binary sensor values, lines of ASCII text, and so forth. The receiving software will treat these messages differently; recall what happens when you display binary data on your PC's screen for an example of what happens when a program processes inappropriate data.

Because these "overhead" codes are required regardless of the actual message data, it makes sense to gather them into a header that precedes the data. The combination of header and data bytes is sent and received as a unit by the network firmware, but the application programs are responsible for creating and consuming the data bytes.

Figure 4 shows the header for messages sent over the INKnet RS-485 network. These bytes provide enough information to route the message, ensure that there are no transmission errors, and keep all nodes informed about the net firmware's status. I will explain more about the header in the next article, but the key point for now is that the shortest network message is ten bytes of header with no data bytes.

The header information requires network bandwidth, of course. Continuing the example from above, the header requires 2% (10/500) of the bandwidth for each message per second. If the messages contain no additional data, a 5000-bps network can handle up to 50 ten-byte messages per second.

Because the network bandwidth is fixed by the serial port data rate, the network can handle fewer messages as they become longer. A line of BASIC code may contain 20 characters, so the complete message has 30 bytes and will occupy 6% of the bandwidth per second. This implies that the net can handle about 17 such messages each second, regardless of their source.

Longer messages use the network bandwidth more efficiently because the ratio of header bytes to message

The first 10 bytes of each message are reserved for use by the network control software. These bytes are:

Byte	Function
00	Reserved, must be zero
01	Message length
02	Target node address ("Sent to")
03	Opcode
04	Source node address ("Sent from")
05	Reserving node address ("Reserved by")
06	High byte of status flags
07	Low byte of status flags
08	High byte of checksum
09	Low byte of checksum

The length byte indicates the total number of bytes in the message. The minimum value is 10 decimal, which will occur when the message has no additional data bytes.

The checksum is the negative sum of the other bytes treated in pairs as 16-bit quantities: even numbered bytes are high-order, odd numbered bytes are low-order.

Figure 4—*INKnet message header*

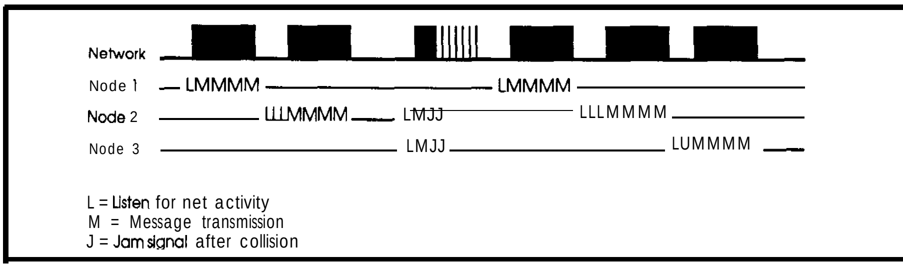
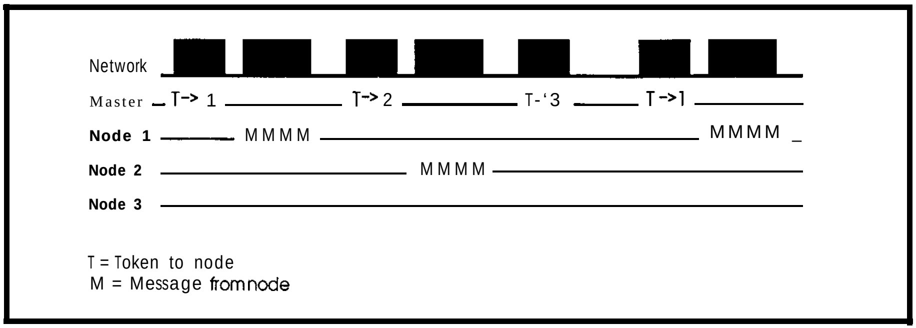


Figure 5—*CSMA/CD network activity*



gure 6—*Centralized token network activity*

bytes drops. There is a tradeoff between **message length,** message frequency, and number of transmitting nodes that must be analyzed for each network application.

But, before I can describe that tradeoff, we must answer the classic question.. .

WHO'S ON FIRST?

Recall that RS-485 allows only one node to transmit at a time without specifying exactly how to achieve this goal. The method has a much larger effect on bandwidth than you might imagine, so it is worthwhile to explore the implications. The buzzword "multiple access" applies to nodes connected to a common wire, be that wire coaxial cable, twisted pair, or whatever.

When two transmitters are active at once, a "collision" occurs between the two messages. Because the messages contain different data (the source nodes and checksums, at least!) and did not start at exactly the same time, the voltages on the network produce garbled data in all receivers. The entire message may be scrambled, but the firmware can easily tell that the result isn't valid.

A node can avoid most collisions by simply not transmitting when another node is using the net. This is called "carrier sensing" because the receiver must be able to detect when the network is in use. The original research into this typeof networkused aradio-frequencycarrierand thename lingers, long after the RF has faded.

However, two nodes may listen at the same time, decide that the net isn't in use at the same time, and begin transmitting at the same time. The resulting collision will be detected if the transmitting nodes compare incoming and outgoing data bytes; any mismatch indicates a collision. This is called, naturally enough, "collision detection."

The two (or more!) nodes participating in a collision force a unique "jam" signal on the network to ensure all other nodes recognize the collision and discard the message. They then wait for a short time and begin listening all over again.

If the colliding nodes wait for exactly the same amount of time, they willcollideagainandagain. Thenodes can prevent this by choosing a random delay based on the node address, the user ID, or any other value that will be different from node to node. The backoff value must be changed to avoid giving one of them priority access after each collision. This is known as a "random backoff" delay.

Ofcourse,anothernodemaybegin transmittingwhile thecollidingnodes are backing off. This means those nodes who both attempted to be on first are now delayed until the end of the new message, at which time they (and any other nodes) attempt to be on first again.

This entire collection of procedures is known as the "CSMA/CD with Random Backoff" protocol (spelled out, that reads Carrier Sensing Multiple Access with Collision Detection and Random Backoff), which may be more familiar when pronounced "Ethernet." There are several Ethernet variations, but CSMA/CD is the most common. Figure 5 illustrates how this works with three nodes contending for network access.

Inherent in the collision detection strategy of a CSMA/CD network is a performance decrease as the number of messages increases. A simple thought experiment illustrates the problem.

Assume that the nodes use 75% of the bandwidth for messages. A node thus has a 75% chance of detecting a message in progress when it wants to send; it can start transmitting only 25% of the time. Under these conditions, several nodes will begin listening to the network during each message. The odds of a collision immediately after that message are quite high because all listening nodes start transmitting at once; all those nodes must delay and retry later.

The collisions and jams occupy network bandwid th, so as the number of collisions increases, the bandwidth available for messages decreases. Under extremely heavy loading, a CSMA/CD network can degenerate to a brawl allowing no useful data transmission at all.

In the simplest example, there is no way to ensure that a node will ever get a chance to send its message. It could collide every time it starts to transmit and lose every random backoff. This problem only crops up under heavy loading, but that is precisely the time when a controller will attempt to send a critical error messages that absolutely must get through!

In any event, the key conclusion is that CSMA/CD does not use network bandwidth particularly well under heavy loading conditions. Unfortunately an RS-485 network will almost always be heavily loaded because the total bandwidth is limited to standard serial data rates. As we saw above, 17 30-byte messages were enough to fill a 5000-bps network to capacity without worrying about contention for network access, the resulting collisions, and the inherent loss of capacity.

But if CSMA/CDisn't theanswer, perhaps we must rephrase the question:

WHO'S ON NEXT?

Another method of avoiding collisions is to allow a node to transmit only when it receives a message from another node containing explicit permission. This type of control is called "token passing" because the node must have a "token" before being granted admission to the network. Unlike the New York City subways, there is only one token available, so only one node may transmit at any time.

Contrary to popular belief, a "token passing" network is not synonymous with a "token ring" network. A "ring" network arranges the nodes in a circular daisy chain with the last node connected to the first. Each node transmits to the next node in the chain and listens to thepreviousnode. While I won't describe ring networks any further, it is worth noting that the ring depends on all the nodes; one dead node disables the entire network.

Al though token-passing networks eliminate collisions, they use bandwidth to transfer the token from one node to the next. There are basically two ways to control the token: by decentralized control distributed among all the nodes, or from a centralized network master node.

Decentralized control simply means that the node holding the token decides where to send it next. A simple algorithm might send it to the next highest numbered node; the last node (the one with the highest node number) sends it to the first node, and so on. More complex algorithms would take into account node priorities, usage patterns, and so forth.

The most significant problem with decentralized control occurs when the token disappears during a node failure or a noise burst. Because none of the remaining nodes have permission to transmit, recovery requires deciding which of the remaining nodes should create a new token and restart the network.

Centralized control implies that a single master node is responsible for shepherding the token among the active nodes. Although it may seem that the node with the token must return it to the master, an easier method is to restrict the active node to a single message (or a few messages) after receiving the token. The token "disappears" after the node transmits its message and the master creates a new one to give the next node permission to transmit.

The advantage of this method is that the master node can maintain a table of active nodes and decide how to schedule future token activity. The disadvantage, of course, is that if the master node fails the network will stop running.

Figure 6 shows the traffic for a network using centralized control. Notice that Node 3 does not respond to the token, so the master **sends a** new token to Node 1 immediately after it determines Node 3 is inactive or missing.

These token-passing schemes do not match the familiar IEEE and IBM Token Ring network protocols. The reason is simple: Token Ring nodes require a full-length PC card with a separate processor, several VLSI network control chips, and a goodly dollop of RAM to manage the interface independently of the host PC. While this provides a high-level interfaceand a high-speed network, it is not relevant for simpler controller networking problems.

PACKET PERFORMANCE

Although network bandwidth determines the theoretical upper limit for data transfer, the actual limit is often imposed by the nodes. You may be surprised to find out how little information your programs can handle!

For example, suppose that an 8052-BASIC board is receiving 250-byte messages and storing them for later use. Moving data from one part of external memory to another is not one of the 8052's strong points, even though it runs at about 1 MIPS, so the node cannot accept another packet for about 4 milliseconds.

Application processing may be even slower. While the BASIC-52 language is well suited for controller applications, it executes only about 1000 statements per second. If your application requires 100 instructions to process a message, it can handle about 10 messages each second.

The secret to effective network use is to reduce the amount of information by processing it within the originating node. If a node is collecting data, report only the minimum and maximum values every minute instead of sending a torrent of raw data. Rather than sending a status message ten times a second, send it only when a change occurs.

Further, your program must respond quickly to incoming messages, because the network cannot deliver another message until your code handles the previous one. You can imagine the throughput effect of a 10-second delay loop embedded in your code.. .

NETWORKING FUTURES

Now that the background is out of the way, the next article will describe INKnet itself. INKnet is an RS-485 network running at 19,200 bps, designed to handle the normal communications load of small control systems using 8052 controllers. A companion program for IBM ATs provides BASIC consoles, supports file transfers, and displays network debugging information.

INKnet uses the centralized control model described above, so a single node is designated the "net master" and handles all of the token logic. The firmware handles all of the network functions, so you can run a normal data collection program on that node; the net does not require a dedicated node for network control. ✣

Downloadable files for INKnet will be available on the Circuit Cellar BBS when the next article appears.

Ed Nisley is a member of the Circuit Cellar INK engineering staff and enjoys making gizmos do strange and wonderous things. He is, by turns, a beekeeper, bicyclist, Registered Professional Engineer, and amateur raconteur.

IRS

210 Very Useful
211 Moderately Useful
212 Not Useful

CEBus: A New Standard in Home Automation

by Ken Davidson

The First In-Depth Technical Description

Whenever the media or popular press write about home control, the story always reads the same. "John is awakened at 6:30 A.M. to the sound of a pleasant voice reminding him of the day's events. The coffee has started brewing and the heat is being turned up in the rest of the house. After a leisurely breakfast, John leaves for the office and the home automatically arms the security system.. ."

Then the article goes on to extol the wonders of home control, that it will be the panacea for stoves left on, groping in the dark for light switches, and high heating bills, without ever saying anything useful about what's available or how it works.

Little has actually appeared in the home control arena to make the life of the typical homeowner easier. True, numerous stand-alone devices have shown up to control isolated aspects of home life, and there have always been the custom installations done for the rich and esoteric that make the house resemble the starship Enterprise, but nothing has appeared geared to the masses for affordable whole-house control. What's needed is a standard to unify all the appliances, consumer electronics, and security devices in the house so they can talk to one another economically.

The EIA (Electronic Industries Association) has taken the lead with the development of CEBus, the Consumer Electronics Bus. With a committee made up of such major players as Sony, Philips, Panasonic, General Instrument, AT&T, Texas Instruments, Mitsubishi, RCA, and Johnson Controls, a standard has been developed to allow all home appliances to communicate over various media.

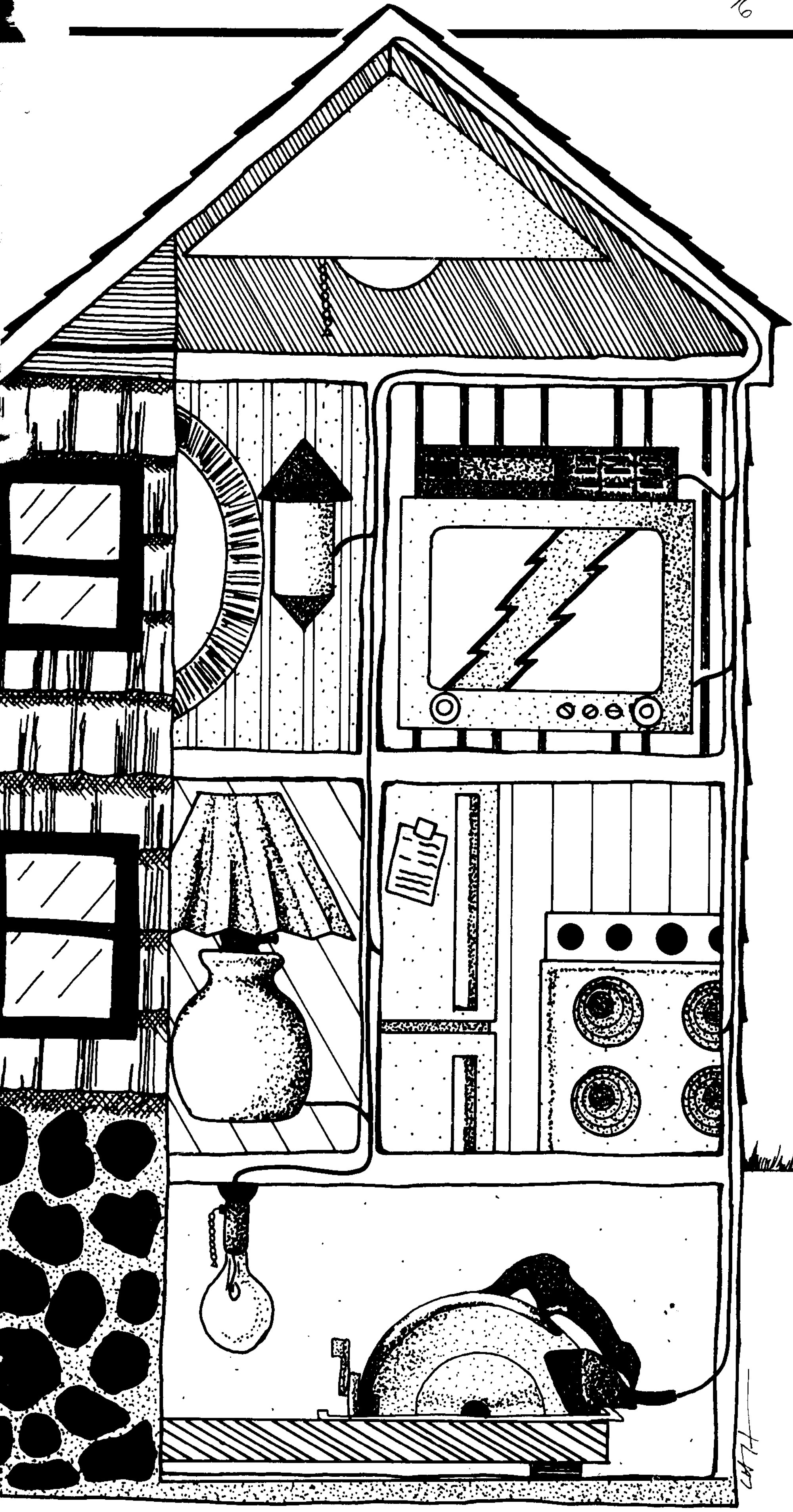
Even though this article contains technical details of CEBus that, up to now, have been available only to committee members and individuals closely tied to the development of the standard, one bit of warning is in order to those who think the information here is the final word: The CEBus specification is still under development and hasn't been formally released. While, at the time of this writing, EIA plans to formally release the specification in November, they are still free to make changes up to that time. The information in this article will help you become familiar with the details of CEBus, but shouldn't be used as the basis of designs.

CEBUS

In 1984, EIA formed the CEBus committee to develop a standard to facilitate communications between various home automation devices and appliances. They started out working on unifying infrared hand-held controllers in an effort to reduce the jungle of remotes found in many entertainment rooms, but quickly discovered that it made sense to extend the standard to whole-house communications over assorted media.

At the time, General Electric was developing their HomeNet system to allow power line communication in the home. (HomeNet is different from GE's X-10-based HomeMinder.) The committee used the protocols developed for HomeNet as the basis for their early standards work.

"Home Automation" has been a buzz phrase of the eighties. There's been a lot of hype promising HAL-like computers taking control of every aspect of our home life, from wake-up to bedtime, making our life just a little less traumatic.



According to EIA documents, the committee had five primary goals for CEBus: It would be retrofittable, use distributed intelligence (have no central computer in order to operate), be non-product specific, have an open architecture, and be expandable. As you'll see from the details that follow, they've been able to achieve these goals quite adequately.

CEBus isn't actually a bus, but a network specification. It follows the ISO/OSI seven-layer network model, which defines the physical, data link, network, transport, session, presentation, and application layers (Figure 1). Each layer is responsible for one aspect of network communication, with each layer only able to talk to the layers directly above and below it. For example, the physical layer is only concerned with getting bits from one node to another, without regard for what the bits mean or even whether they make it from one node to the next error free (error detection and correction are handled by the data link layer, which is one level higher).

By breaking the model into well-defined pieces, implementation and support are greatly simplified. It's also possible for one company to implement specific layers, with another company implementing the rest. The two implementations talk to each other through a well-defined boundary that exists between the layers.

PHYSICAL LAYER

At the lowest level is the physical layer. This is where CEBus's greatest strengths lie since several different media are defined in the specification, with the choice of which medium to use up to the appliance designer. All the layers above the physical layer are identical regardless of medium, so the network is medium independent.

Signaling is done on most of the media by switching between a "superior" state and an "inferior" state. Times between changes determine the information being conveyed. "One" bits last one "Unit Symbol Time" (UST), "zero" bits last two USTs, end-of-field markers last three USTs, and end-of-packet markers last four USTs.

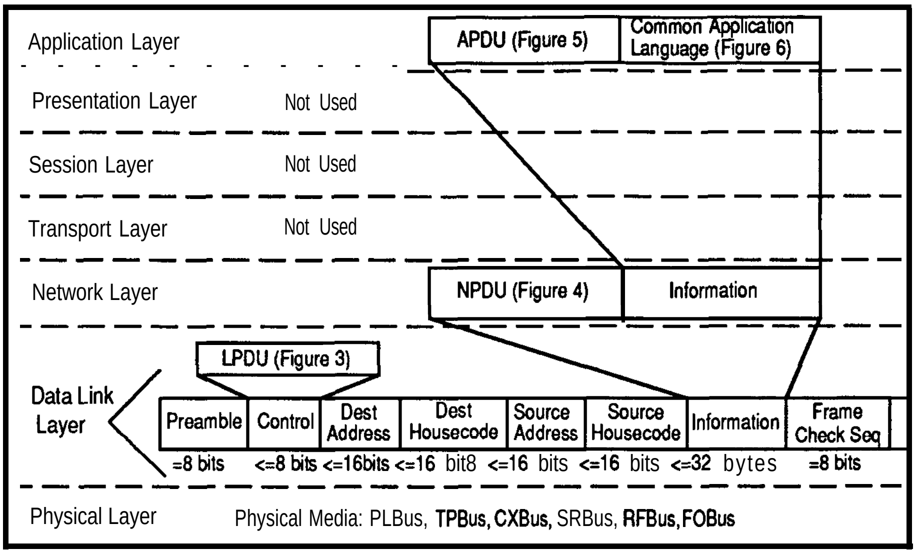


Figure 1—*CEBus is based on the ISO/OSI seven-layer network model.* **Each of the** *layers is responsible for just one aspect of the network connection.*

Exactly what defines the superior and inferior states dependson the medium. Also, since characterizing communication speed for a medium in bits per second is meaningless since one bits and zero bits are of different durations, data rates are usually defined in terms of "one bits per second." Statistically, the overall throughput in bits per second is around two-thirds the value of one bits per second.

The CEBus specification defines six media which may be used to carry the signal: PLBus (Power Line Bus), SRBus (Single-Room Bus, or infrared), RFBus (Radio Frequency Bus), TPBus (Twisted-Pair Bus), CXBus (CoaX Bus), and FOBus (Fiber-Optic Bus), the last three of which are often collectively referred to as WIBus, or WIred Bus.

PLBus

PLBus is likely to be the medium of choice for most appliances meant for retrofit installations since almost every house and business in the world is wired for electricity. Since the power line is such a harsh environment, with noise and transients the norm, this is the slowest of all the media, but is still able to attain a data rate of 1000 one bits per second with a UST of 1 ms.

Transmissions use a 120-kHz carrier to denote a superior state and the lack of a signal for an inferior state. Unlike the X-10 system which transmits only at the 60-Hz zero crossing, PLBus transmits regardless of the state of the AC power on the line. As a result, transmission can still take place even if power isn't present, something that can't be done with X-10. And even though both PLBus and X-10 use 120-kHz carriers, the two systems are completelyincompatibleand,indeed, interfere with one another (see the sidebar "CEBus and X-10: Harmony or Discord?").

SRBus

SRBus is an attempt at eliminating the plethora of infrared remote controls that litter any well-stocked entertainment room. You should be able to have a single hand-held remote that transmits all valid CEBus commands. Not only will you be able to talk to the television or VCR in the same room (even if they are made by different manufacturers), but with the proper bridge in place to retransmit the SRBus signals onto PLBus or one of the other media you should be able to control any CEBus-compatible device, including the lights all over the house or the door opener out in the garage. (See the sidebar "A CEBus Demonstration" for an example of just how powerful the hand-held infrared remote can be with CEBus.)

SRBus uses a 40-kHz infrared carrier and pulse-position signaling to attain a data rate of 1000 one bits per second. A 50-μs burst of IR is used to indicate a transition from superior to inferior and from inferior to superior. By using just short pulses, the hand-held remote's battery life is extended.

RFBus

Currently used predominantly in the security industry, RF is another medium that would work well in retro-fits. FCC regulations limit the strength of RF transmissions, so whole-house coverage may be possible without interfering with the neighbors' CEBus appliances, Environmental interference and limited range could hamper the usefulness of the medium.

Little has been published regarding the technical aspects of RFBus, so there is little to tell here. Presumably, the details will be worked out by the time the specification is formally released, though EIA may opt to delay formalizing the RFBus portion.

TPBus

TPBus promises to be the most useful high-speed medium in the majority of installations. While most houses don't have an abundance of spare twisted-pair wire runningroom to room, some may have extra telephone pairs that could be used in retro-fits. Twisted pair is cheap enough that many home builders may opt to install it from room to room along with the AC and telephone wiring at the time of construction.

TPBus runs at a data rate of 10,000 one bits per second and uses a ±125-mV peak-to-peak signal. While such a small signal amplitude may seem prone to picking up noise, the committee was concerned about preventing noise from being radiated onto audio lines that may be run next to the control lines. Similar to SRBus, TPBus uses 50-µs pulses to indicate transitions from superior to inferior and vice versa. Pulses alternate between going from ground to +125 mV and from ground to -125 mV.

CXBus

With the spread of cable TV, many existing homes and more and more newly constructed homes are being wired with coax cable for television distribution. Since, within thecontext of the house, the TV signal isn't using theentirebandwidthofthecable, there is plenty of room for adding control information plus high-quality audio and video to the same cable.

CXBus consists of a pair of coax cables, one for upstream feeds and one for **downstream** feeds. The downstream signal is used to feed the inputs of devices and uses active taps to extract the signal. The outputs from devicesare passively combined on the upstream line. The upstream signal feeds into CXBus node 0, where it is combined with any incoming cable TV or off-air signals and is used to drive the downstream line. Node 0 also contains a CX/TP bridge.

The frequency allocation for CXBus is shown in Figure 2. The specification will eventually have provisions tomodulatein-house video signals onto unused TV channels for reception by TVs on the downstream side. UHF channels 62-69 will probably be used for this purpose.

CXBus uses the same pulse-width modulationused by PLBus, but with a UST of 100 µs, providing a data rate of 10,000 one bits per second.

FOBus

Fiber optics are becoming the medium of choice where high data transmission rates and low noise pickup are important. While some provisions have been made for this medium in the CEBus protocol definitions, very little work has been don

CXBus Upstream Frequencies	
0-0.2 MHz	Control Channel
5-47 MHz	Analog Data
54-88 MHz	In-House-Generated Video

CXBus Downstream Frequencies	
0-0.2 MHz	Power
5-47 MHz	Analog Data
54-88 MHz	In-House-Generated Video
88-100 MHz	FM
108-216 MHz	Off Air

igure 2—*CXBus mixes control, analog, and video on the same cable.*

on the physical details. The specification will likely be released initially with a large hole where FOBus is supposed to go, but will probably be filled in once the rest of CEBus has a foothold in the industry.

DATA LINK LAYER

The next highest level is the data link layer. It is primarily responsible for providing a clean channel of communication for the higher levels. To do this, it must handle collision prevention, detection, and resolution; packet acknowledgment; and final packet construction.

Collision prevention, detection, and resolution is handled using CSMA/CDCR (Carrier Sense, Multiple Access/with Collision Detection and Collision Resolution). Since all nodes are connected to a common medium with no master node dictating who may transmit when, there always exists the possibility that two nodesmaytry transmittingat thesame time. Since it is always best to avoid such a situation, collision prevention is tried first. *[Editor's Note: For a more detailed discussion of networking in general, see "A Network for Distributed Control" on page 32.]*

Before transmitting, each node listens to the network to determine if anyone else is already transmitting. If so, it simply waits for the transmitting node to finish. When the network is free, the node waits a certain amount of time before trying to transmit. The time it waits depends on the packet's priority (high priority packets are tried sooner than low priority packets) plus a random channel access delay. Without the randomizing factor, if two nodes see the network become free at the same time and have packets to send of equal priority, both will try transmitting at the same time, resulting in a collision.

When a node determines that it is going to start transmitting, it starts by sendingout a preamble character. The preamble is a random number designed to be a "sacrificial lamb." The transmitting node listens as it sends out the preamble, and if the preamble survives intact, the rest of the packet is sent. If a collision is detected, transmission is aborted and the process starts again.

Depending on the packet type, an immediate acknowledge may be requested by the sending node. After transmission is complete, but before the sending node gives up the communications channel, the receiving node will send an acknowledgment back to thesender. If thesenderdoesn't receive the acknowledgment before timing out, the packet isretransmitted once. If there is still a problem, a higher network layer must decide what to do.

Thebytesentafter thepreambleis the Logical Link Control Protocol Data Unit, or LPDU (Figure 3). It contains the packet type (which determines acknowledged or unacknowledged service and local or nonlocal medium), packet priority, privilege, and basic or extended service.

Following the LPDU are the destination address, destination house code, sourceaddress, and source house code. Each node is identified by both a unit number and a house code. It's possible to have several independent networks using the same media by assigning different house codes to different groups of devices. It's also useful for avoiding conflicts with neighbors (especially in the context of PLBus and RFBus).

Both destination and source addresses are sent so the receiving node knows the sender's address. For example, there may be several TVs and a single VCR connected to the same CXBus. One of the TVs may send a request to a VCR to start playing a tape, but the VCR must first send a command back to the TV to change to channel 4. Knowing the address of the TV which sent the play command, it's possible for the VCR to tell the correct TV to change channel.

7	6	5	4	3	2	1	0
Serv		Priv	Priority		Packet Type		

Service Class

0	Basic Service
1	Extended Service

Priviledge

0	Nonprivileged LPDU
1	Priviledged LPDU

Priority

00	Deferred
01	Low
10	High

Packet Type

000	Local Data Unacknowledged
001	Local Data Acknowledged
010	Acknowledge Response
011	Nonlocal Data Unacknowledged
100	Nonlocal Data Acknowledged
101	Failure Response

Figure **3**—*The LPDU* describes *several low-*level *characteristics of the packet.*

After the packet addressing is the packet information. The information comes from the higher network levels and will be discussed in more detail in a moment.

The final field contains the Frame Check Sequence. It is simply an 8-bit checksum of all the bits in the packet excluding the preamble.

NETWORK LAYER

The network layer is responsible for determining which media are to receive the packet and for breaking apart packets which would end up being larger than the 32-byte limit.

The Network Protocol Data Unit (NPDU) is added to the front of the information field passed down by the upper levels and is shown in Figure 4. There are six bit fields which determine which media are to receive the packet. Setting a bit in the field results in the corresponding medium receiving the packet (assuming the proper bridge is present to transfer packets acrossmedia). The last two bits determine whether the packet is to be sent using flood routing, directory routing, or directory routing with a request for a return ID, and whether it is being segmented.

In flood routing, the packet is sent to every medium specified in the rest of the field. In directory routing, the packet is only sent to the medium which hosts the destination node. For example, if a packet is specified for coax, power line, and twisted pair, and is to be sent to node 3, the packet will only be sent to the PLBus if directory routing is used and node 3 is connected to thePLBus. Suchascheme relies on the presence of routing tables in the bridges. Should directory routing be requested, but a bridge not haveanentryfor thedestinationnode, the bridge will change the packet to request that an ID be sent by the destination node. Upon receipt, the destination node sends out an ID so that bridgescan update their routing tables for future reference.

7	6	5	4	3	2	1	0
Type		RF	SR	FO	CX	TP	PL

NPDU **Type**

11	Flood Routing, Unsegmented
10	Directory Routing, Unsegmented
01	Direectory Routing, Return ID, Unsegmented
00	Segemented Variable Header

Figure 4—*The NPDU* determines which me*dia are to receive copies of the packet.*

TRANSPORT, **SESSION, and PRESENTATION LAYERS**

Since theOSI seven-layer network model was designed to be useful in just about any application, there is bound to be some fat that can be trimmed while implementing applications that don't require all the facilities or segmentation defined in the model. Such is the case in the definition of CEBus. The functions of the transport, session, and presentation layers as defined in the OSI model are handled by the application, network, and data link layers in the CEBus definition. This doesn't mean that the OSI model isn't being followed or that corners are **being** cut; there are merely certain facilities found in larger networks that just don't exist in a simple control network.

APPLICATION LAYER

The highest level is the application layer and is responsible for what the end user ultimately sees. In the case of CEBus, the highest level defined isn't what the end user will see (because, in many cases,operation will be transparent or part of a device's existing functionality), but what the programmer sees. EIA has defined CAL, Common Application Language, to allow devices to communicate intelligently with each other.

Before getting into CAL, though, there is a header similar to those found in the lower layers that is added to the front of the CAL command before being passed along called the Application Protocol Data Unit (APDU). The APDU may be up to 3,810 bytes long, but only the first two bytes have been defined at present (Figure 5). The first byte contains the mode information and type identifier. The mode specifies the service class, header type, and data filed length for the command which follows. The service class may be either basic or privileged, though most commands in use at this time are basic. The header may be either fixed or variable in length, with fixed-length headers being the norm. Finally, the command length may be either short (up to 32 bytes), long (up to 3,808 bytes), or huge (up to 1,638,375 bytes).

The type identifier determines whether command is implicit or explicit, and defines the response codes for an explicit the command. An implicit command doesn't require a response, so is simpler to program and faster to send, but is subject to

Call for Manuscripts

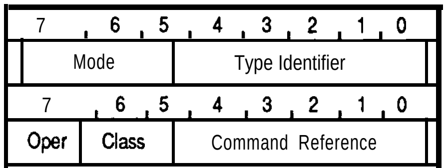
CIRCUIT CELLAR INK is looking for quality manuscripts on software for embedded control, software applications, advanced algorithms, software methodology, and tutorials on tools and techniques for developing software.

These manuscripts will be considered for publication in CIRCUIT CELLAR INK, THE COMPUTER APPLICATIONS JOURNAL, and in a planned series of books to be published by Circuit Cellar INK.

CIRCUIT CELLAR INK offers writers and engineers a technically sophisticated audience and professional editorial guidance.

The CIRCUIT CELLAR INK Author's Guide is available for downloading from the Circuit Cellar BBS. Prospective authors may send mail to 'Curt Franklin' on the Circuit Cellar BBS, or send proposals for manuscripts and requests for Author's Guides to:

Curtis Franklin, Jr.
Editor in Chief
Circuit Cellar INK
4 Park Street
Vernon, CT 06066



Mode
000 Basic service, Fixed header, Short data
001 Priviledged service, fixed header, Short data
010 Basic service, Variable header, Long data
011 Priviledged service, Variable header, Long data
100 Basic service, Variable header, Huge data

Type ID
01011 Explicit Association Invoke
01100 Reject
01101 Error
01110 Result
01111 Implicit Association Invoke

Operation
0 Asynchronous Transmission
1 Synchronous Transmission

Class
00 No response requested
01 Error or Rejected requested
10 Result requested
11 Result, Error, or Reject requested

Command Ref
User-definable codes (except 00000)

Figure 5—*The APDU is used to convey error information as well as type of service and communication.*

errors since thedestinationnodenever gives a response. An explicit command requires a response from the destination node, with the response either a result, reject, or error code.

The second byte in the APDU determines whether transmission will be synchronous or asynchronous, and what kind of response isdesired in the case of an explicit command.

CAL is made up of three sections, with each section more precisely definingjust what thefinalactionshould be. The first section is the context and defines general categories for devices. For example, there are categories already defined for an audio process (amplifiers, loudspeakers, equalizers, etc.), video monitor, communication control system (telephone, radio, etc.), time service element (real-time clock, timer start and stop times, etc.), environment management system, lighting system, and the list goes on.

The second section is the Specific Application Service Element,or SASE. The SASE defines the primary function of the command sequence. Each context has a list of SASEs defined, and are often similar or identical across differentcontexts. ExamplesofSASEs for, say, an audio process are primary mode switch (power), source switch (radio, CD, or tape), feature switch (noise reduction, surround sound), and level controls (volume, bass, treble, balance).

Finally the Common Application Service Element (CASE) defines just whatthefinalactionshouldbe. CASEs are thesameforallcontextsandSASEs. Example CASEs include true, false, add, subtract, and load.

Using the context, SASE, and CASE, it's possible to create commands to do just about any function you can think of. EIA has tables of predefined contexts, SASEs, and CASEs, so most of the time it's just necessary to look up a command in the table and use it. Manufacturers who want to add commands can follow the rules for CAL and develop new commands. Escape codes have been put in place to allow unlimited extension of CAL commands should the main tables ever fill up. EIA presumably will have ultimate control over the tables and the commands that go in them. Possibilities exist for manufacturers to implement proprietary command sequences, but it's unlikely such a product would win any sort of official CEBus-compatible approval.

Figure 6 shows a sample of what has been defined for CAL so far. As an example, suppose you want to turn a TV on. The context would be 38 hex to select a video monitor, theSASE would be DO hex for primary switch, and the CASE would be 81 hex for true, or on. The final command would be 38H, DOH, 81H.

HARDWARE

Even though the CEBus specification isn't slated for formal release until November, several companies sitting on the committee have started work onhardwareforimplementingCEBus functions on single chips to make manufacturers' jobs easier when adding CEBus support to their products.

CyberLYNX Computer Products, working with Texas Instruments, is developing a single-chip CEBus interface that will handle all the details of CEBus communications. Though the chip is still in its early stages, Cyber-

a) Context	Hex Value	Examples
Audio Process	30	Amplifiers, equalizers, noise reduction, etc.
Video Monitor	38	TV sets and computer monitors
Tuning System	40	Radio, TV, VCR, cable box, etc.
Appliance Control System	60	Dishwashers, washing machines, etc.
Environment Sensors	69	Temperature, humidity, pressure, etc.
Security System	74	Sensors, alarms, etc.
Lighting System	78	Lights of all kinds

b) SASE	Hex Value	Description
Primary Mode Switch	D0	Main functions such as power
Source Switch	D1	Input selection

c) CASE	Hex Value	Description
ADD	86	Add values together
NOT	93	Toggle item
TRUE	81	Turn item on

Figure 6-Examples **of contexts** *(a), SASEs (b),* **and** *CASEs (c) show the* **potential** *CAL holds for assembling comprehensive commands.*

LYNX has a CEBus evaluation board that implements many of the CEBus functions in firmware. It also includes a power line interface so several boards can communicate. A block diagram is in Figure 7.

The functionality of the firmware on the evaluation board is on the primitive side, and the documentation is sparse, but the board allows a developer to try out CEBus with minimal effort.

Also working on a single-chip CEBus interface is AISI Research Corp. At this time, they appear to have a jump on CyberLYNX/TI since they are actually shipping early versions of their chip on evaluation boards (Figure 8) along with some nice-looking software and documentation. The AISI CEBus implementation is dubbed SPIRIT and takes the idea of CEBus control one step further than most would people would think. Rather than provide a chip that only handles low-level network details (similar to the way the NCR5380 provides a low-level interface to SCSI), the SPIRIT chip actually contains some intelligence and is capable of being trained and can operate stand-alone without additional processor power.

The SPIRIT chip has eight discrete inputs, eight outputs, microprocessor control lines, and the requisite CEBus input and output connections. Even though development of the actual 40-pin chip hasn't been completed, AISI has developed an intermediate 18-pin custom chip that handles all the network timing and has put it on a small board with a processor and a 40-pin header that emulates the final 40-pin chip. They have put this emulator board on top of one of their power line modem boards to create a develop ment kit that provides hardware functionality identical to what will be available when the final 40-pin chip is finished.

In its simplest configuration, the chip is capable of monitoring network communications and changing output bits based upon received commands, and monitoring input bits and sending out commands based on those inputs. The chip contains several hundred bytes of EEPROM and is trained before hand with which CAL commands to watch for, what outputs to change upon receipt of those commands, what inputs to watch for, and what commands to send out in response to those inputs. Once trained, the chip

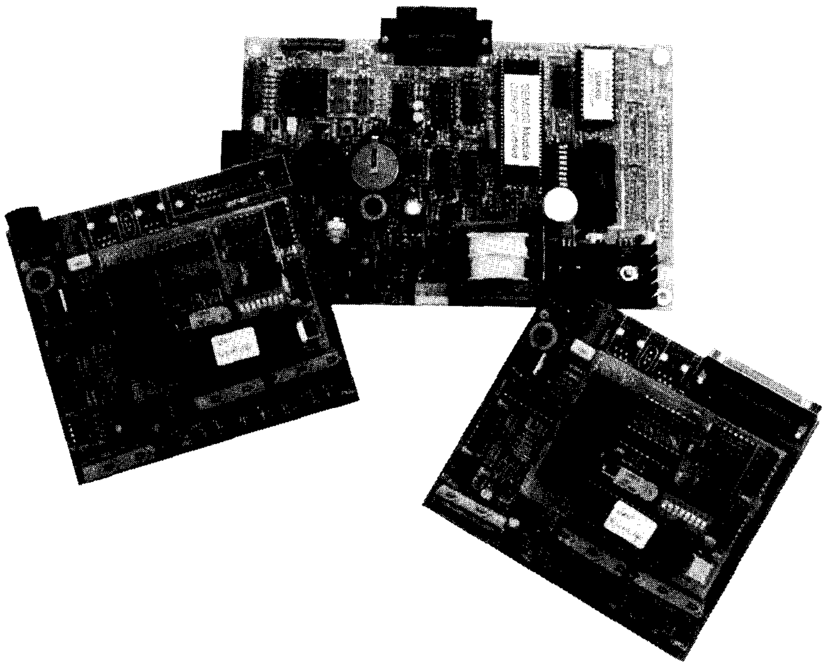


Photo 1—*Clockwise* **from** *the lower right are AISI's power line computer interface, CE200 module, and CyberLYNX's SEM200.*

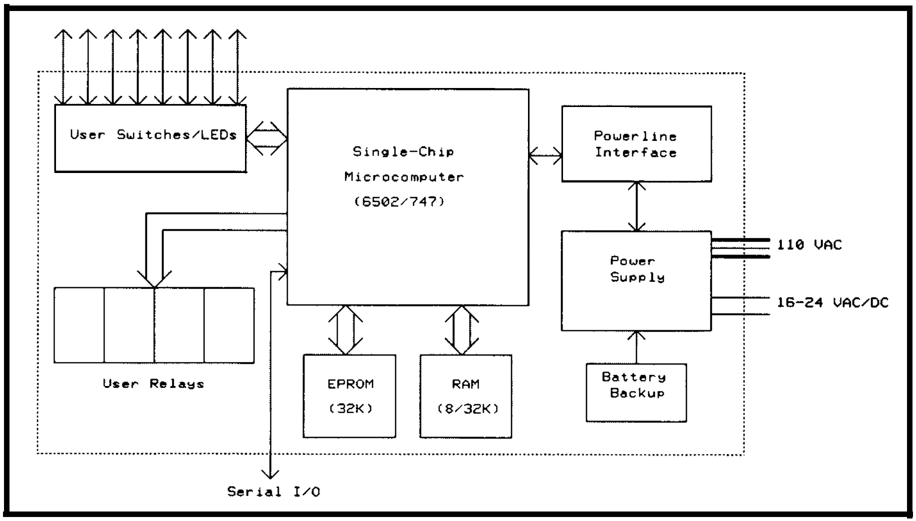


Figure 7-The *CyberLYNX SEM200 implements all CEBus functions in firmware rather than hardware. As a result, functional updates may be made quickly and easily as the CEBus specification is fine-tuned.*

plus any interface circuitry will operate stand-alone. For example, you can train the chip to turn an output bit on upon receipt of a "Light On" command and turn the bit off upon receipt of a "Light Off" command. Additionally, you might add some interface circuitry so that an input bit goes on whenever the light is on and that goes off when the light goes off. It's possible to train the chip so that it verifies that the light goes on when it's supposed to and similarly goes off when it's supposed to and sends out either a confirmation or an error message. Put the chip and interface circuitry into a small module and it will operate without any additional smarts.

In cases where additional processor power is necessary, the chip can also be connected serially, or right to the microprocessor's data bus. The chip's hardware has been designed to allow numerous modes of operation depending on how it is hooked up.

Preliminary software to allow training of the chip is fully menu driven. AISI's current development kit contains two stand-alone modules with PLBus interface and one module for connecting an IBM PC to the power line. DIP switches set each module's node address and all training is done with the IBM PC through the power line (no direct connection is ever made between computer and modules). A test mode allows the operator to send commands to each module and displays responses so training may be checked out.

Since the CyberLYNX and AISI evaluation boards purport to be valid CEBus implementation and both have power line interfaces, the natural next step was to connect them both to the power line to see if they would talk to each other. The CyberLYNX board

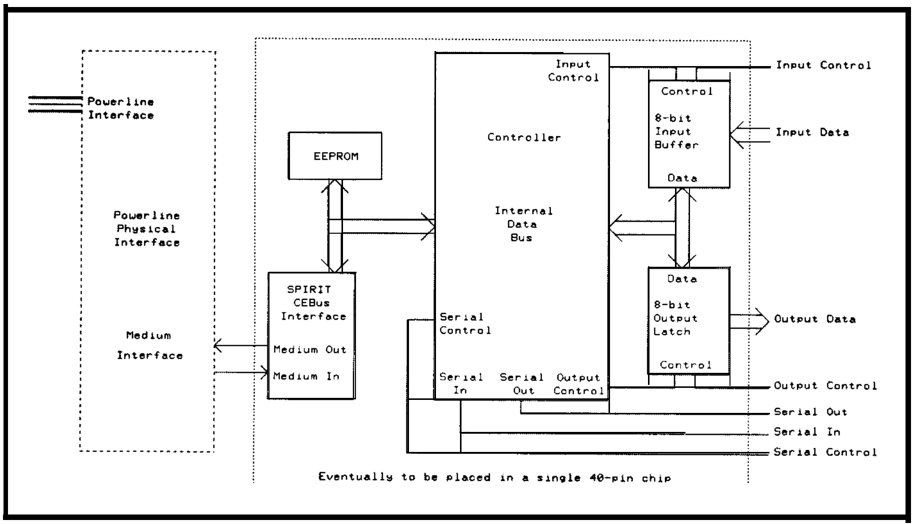


Figure 8—AISI's SPIRIT interface consists of a custom chip for CEBus timing with a microcontroller to supervise. All the CEBus interface circuitry will eventually be contained in a **single** *40-pin chip.*

has a mode in which it will monitor the power line and report back statistics such as number of valid packets, partial packets, bad packets, packets meant for other nodes, and so on, that is has received. It will also count the number of fragments it receives. Putting it into this monitor mode and doing some activity between the AISI modules should result in the count of valid packets received by the CyberLYNX board to go up. However, the only count to ever change was the number of fragments received, which indicated that the CyberLYNX board saw the activity, but couldn't interpret it as valid CEBus code; it only picked up bits and pieces.

Now before you condemn either company, realize that such problems aren't really a bad thing. The CEBus specification is still in its infancy and the evaluation boards may well be based on different, and slightly incompatible, stages of development. Both boards are identical at the upper levels of the network model, which is where most, if not all, developers are concerned. And since each company's boards work with other boards from the same company, that should be sufficient for early development work. As the CEBus specification is refined and finalized, the low-level details of the two companies' boards should become better aligned and complete compatibility should fall into place. For the moment, both boards provide an excellent means for someone to get familiar with CEBus in preparation for final releases.

RETROFIT OR NEW CONSTRUCTION?

A question often raised **when** dealing with home control issues is whether the specification is designed for use in retrofitting existing houses or is meant for new construction only. Retrofitting existing houses is by far the largest market for such devices and certainly can't be ignored. Indeed, the success of the X-10 system has been its perfect match to retrofit applications since no new wiring **must** be installed.

When defining CEBus, the retrofit market certainly hasn't been ignored. Since no wires are needed for SRBus and RFBus, both lend themselves to easy inclusion in any retrofit project. And since just about any house contains AC wiring, PLBus will probably be the most popular medium for retrofits. EIA does recommend the installation of a bridging capacitor between the two 110V legs found in most houses to allow house-wide transmission, but people have been doing that for years in X-10 installations.

While some houses may have extra telephone wire or coax (for cable TV) in place, TPBus and CXBus installations are best done in new construction. Obviously, the other media may

CEBus and X- 10: Harmony or Discord?

Since its introduction in the late seventies, the X-10 power line control system has become the de facto standard in carriercurrent communications. All successful home control systems marketed in the last few years have had some degree of X-10 compatibility. Sears, Radio Shack, and Leviton have been selling X-10-compatible products for quite a while, and Heath and Stanley have recently jumped on the wagon. Now that CEBus has come along with a method of communicating over the same power line, where does X-10 fit in?

Bluntly stated, CEBus's PLBus and X-10 are completely incompatiblesystemsand actuallyinterferewitheachother's operation. Future home control systems will have to choose between the two methods if reliable, consistent operation is desired. The incompatibilities arise from differences in both physical and functional operation.

X-10 relies heavily on the zero crossings of the 60-Hz AC power signal. At each zero crossing, a bit is transmitted, with a burst of 120-kHz denoting a one bit and the lack of a burst denoting a zero bit. While there are 120 zero crossings per second, the X-10 protocol states that each bit shall be sent twice for error checking purposes, with the second copy a complemented version of the original, so the effective bit rate is 60 bits per second.

CEBus uses a stream of bits that have no relation to the 60-Hz AC signal, and will actually work without AC present (unless the device trying to send the message is relying on that same AC for its power), with a maximum bit rate of 1000 one bits per second. The issue of interfering with oneanother arises from the fact that both X-10 and CEBus use bursts of 120 kHz for signaling. Sincedevicescompatiblewithoneortheotheraresensitiveto120 kHz, transmissions by one system will interfere with those of the other.

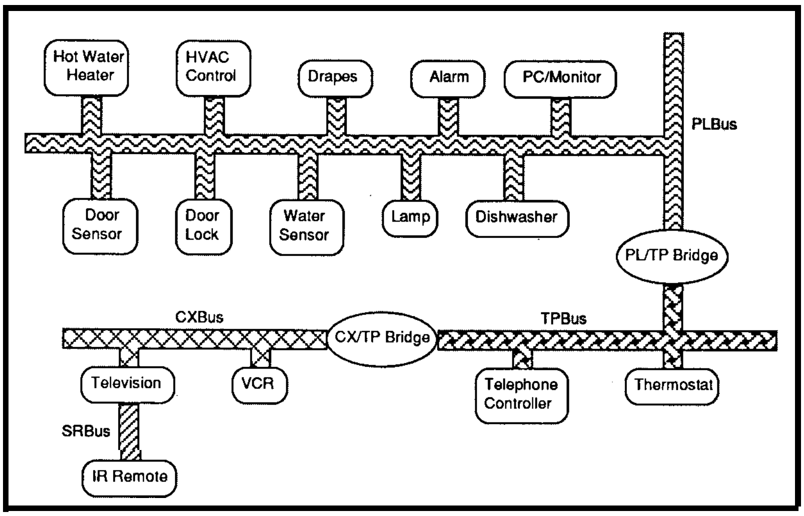
A CEBus Demonstration

CEBus. Yea, we've heard all the hype. How it is going to revolutionize the home by standardizing the interface between appliances in the home. But we here at CIRCUIT CELLAR INK tend not to believe the marketing hype until we can see something concrete, preferably in hand. So when AISI sent us all the information on their CEBus interface products, it was met by a bit of skepticism. Sounds good on paper, but when are we going to see something real?

About the time that we gave up on seeing something work in the flesh, we got a call from AISI about an upcoming meeting of CABA, the Canadian Association of Building Automation. During the course of the meeting, representatives from several Canadian companies involved in buildingautomation were tomeet to discuss standards and developments in the industry, and the hands-on portion of the meeting was to be a live CEBus demonstration.

Not wanting to pass up such an opportunity, Curt and I jumped in the car for the trek to Montreal.

After some opening formalities, we were introduced to Ludo Bertsch, head of research and development at AISI and responsiblefor thedevelopment of AISI's SPIRIT chip, who conducted the demonstration. Figure A shows a diagram of how the equipment was connected for the demonstration. (No, he didn't have a hot water heater or dishwasher. The devices were simulated, but the communication was real.)



Ludo first explained that the entire demonstration would be controlled by a hand-held infrared remote control. (The remote control he was using was the same ONE FOR ALL used with the X-10 IRS43 gateway described in the June/July '89 issue of CIRCUIT CELLAR INK.) The IR remote would send out all the commands to the Sony TV located across the room, which would relay them onto the coax cable delivering the cable signals to the TV. Other devices on the same coax would receive the signals directly, while it would be up to bridges located on the coax to route the signals to the other media.

He first sent the usual commands to the TV, such as channel changing, volume control, and muting. He then turned on the power to the VCR sitting below the TV. While it may not seem like much to turn on the VCR power remotely, realize that the command to turn the power on was sent by the IR remote to the TV, then through the coax cable to the VCR.

Next he told the VCR to start playing the tape. All the usual VCR commands could be sent by the IR remote and relayed by the TV to the VCR via the coax.

Ludo then proposed a scenario in which we're watching the tape and decide the room temperature is a bit too cool. A press of a button on the IR remote brought up a line of text on the bottom of the TV screen telling us that the current room temperature was 68degrees. A coax-to-power-line bridge tookcareof relaying commands between the coax cable and the power line.

Another issue is the overall functionality. The X-10 system was designed for one-way communication. The homeowner has a control console next to his chair which he uses to send commands onto the power line. The remote modules listen to the power line for commands and respond when spoken to. The remote modules have no means of sending their own messages onto the power line for acknowledgements or reporting status, and the command consoles have no way to listen to the power line.

With the recent introduction of the TW523 two-way computer interface, a computer now has the capability to listen to X-10 activity on the power line as well as initiate it, but since the remote modules still have no way to report back errors or current status, the two-way capability is of limited use. X-10, for all intents and purposes, is a one-way, open-loop system with limited potential for intelligent home control.

CEBus, on the other hand, is a completely closed-loop system, with all nodes capable of either transmitting or receiving, and protocols firmly in place that allow acknowledgements and status reporting. Using CEBus, it's possible for a lamp to report back to whomever is trying to turn it on that the bulb is burned out and the light really didn't go on as requested. Using X-10, the controller would have to assume the light went on, without ever knowing if it actually did or not.

X-10 will likely be around for a number of years until more manufacturers begin to incorporate CEBus interfaces into their products and the economies of scale get the prices down. However, because of its faster data rates and true closed-loop operation, CEBus stands to eventually takeover the power line control market.

The Johnson HVAC controller was plugged into the power line and received the temperature request from the television. The controller polled the thermostat located in the room in which we were watching TV and sent a reply to the TV, which displayed the temperature on the screen. The entire request/response time was less than a second.

Remarking that the temperature was too low, Ludo decided to check the current thermostat setpoint. A press of a button on the IR remote resulted in the current setpoint being displayed on the TV screen.

Pressing another button on the IR remote let Ludo bump the setpoint up a few degrees. Within a few seconds, the reading on the thermostat on the wall next to us changed to the new setpoint.

Proposing another scenario, Ludo mentioned that the sun coming in through the windows was a bit too bright and that they would be better closed. Pressing yet another button on the IR remote resulted in the drapes closing. Similar to the HVAC unit, the drape control unit was plugged into the power line. The command was relayed by the television onto the coax cable, which was bridged to the power line.

On to the door leading out of the room. Ludo first requested the status of the door lock, resulting in a line of text on the bottom of the TV screen telling us it wasn't locked. He then sent another command to lock the door, with a confirmation being displayed on the TV when it had been positively locked.

In order to illustrate the closed-loop benefits of CEBus, Ludo proceeded to disconnect the wire running from the door lock controller to the lock itself, then tried to unlock the door. He was still able to talk to the controller over the power line, but the controller was smart enough to sense that the lock wasn't working, so sent an error message back to the TV screen. Rather than our having to hope that nothing went wrong, we got feedback indicating whether or not it locked correctly.

Another example of closed-loop intelligence is a light controller that can sense how much current is flowing through the bulb's filament and will return an error should it decide that the light bulb is burned out.

Ludo also had both a water sensor and a door sensor set up. When either sensor was tripped, a system-wide message was sent out so that everything in the house keyed to respond to such urgent messages would respond. For example, an audible alarm might sound, while the TV might turn itself on and display the alarm condition.

Ludo continued by explaining that CEBus doesn't just make being at home easier. With a telephone interface, control of all "CEButized" appliances in the house can be done over the phone lines. Ludo proceeded to make a phone call to the telephone interface located in the same room. Upon answering, a voice asked for a security code. Once punched in, the interface allowed complete access to the whole house. Ludo was able to turn lights on and off, check current temperatures and setpoints, and start the VCR, all with voice confirmation at completion. The telephone interface was connected to the in-house twisted-pair bus, so a twisted-pair-to-power-line bridge provided the link to most of the devices connected for the demonstration.

At the completion of the demo, most of the CABA members went back to their meeting while Curt and I remained with Ludo to ask more questions. While the hardware is still in the development stages (as is the software and, indeed, the EIA specification itself), final development kits should ultimately provide manufacturers of consumer electronics, appliances, and security systems with powerful and easy-to-use tools for incorporating CEBus interfaces into most of their products with minimum fuss.

also be used, but at the loss of throughput. When installing wire for TPBus (and telephones), EIA recommends the use of 24-AWG copper 8-conductor wire,arranged in four twisted pairs, with one wire run for telephone and another run for TPBus. While loop wiring could be used, they recommend running separate wires from each room to a single distribution panel, **presumably** located in the basement or an attached garage and close to an AC outlet for main power supply and for easy connection to a TP/PL bridge.

Similarly, a pair of coax cables should be run throughout the house (one for **upstream** connection and one for downstream). A loop configuration may be used, or separate wires may be run from each room to a distribution point for added flexibility. The EIA recommendation mentions that one of the wires may be used for conventional cable TV signal distribution until a true CXBus implementation is put in place.

Remember that since the spec is still being developed, these wiring recommendations should be taken as suggestions only. EIA cautions not to "incur additional costs in reliance upon this nonfinal standard."

JUST THE BEGINNING

One stumbling block to worldwide use of CEBus is that the Japanese and the Europeans are both working on home automation standards (HBS in Japan and D^{2}B in Europe). A committee has been formed with members representing all three standards to work out a combination of the three for a true worldwide home automation standard. Such a goal is certainly admirable, and if enough work is done before any of the standards have been set in concrete, it may actually be attainable. It remains to be seen whether the goal is realistic, though.

In any case, CEBus promises to make home control as popular as televisions and dish washers, while as affordable as the current crop of X-10 modules and devices. ✣

Special thanks to Tom Mock at EIA, Ludo Bertsch at AISI, and Les Larson at CyberLYNX for their contributions to this article.

Electronic Industries Association
2001 Eye Street, N.W.
Washington, DC 20006
(202) 457-4900

AISI Research Corporation
2010 Crow Canyon Place, Suite 202
San Ramon, CA 94583
(415) 866-2222

CyberLYNX Computer Products, Inc.
2885 E. Aurora Ave., Suite 13
Boulder, CO 80303
(303) 444-7733

Ken *Davidson is the managing editor and a member of the* CIRCUIT CELLAR *INK engineering staff. He holds a B.S. in computer engineeringand an M.S. in computer science from Rensselaer Polytechnic Institute.*

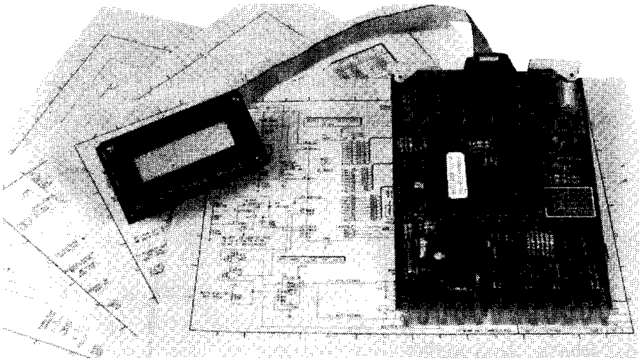
IRS

2 13 Very Useful
214 Moderately Useful
2 15 Not Useful

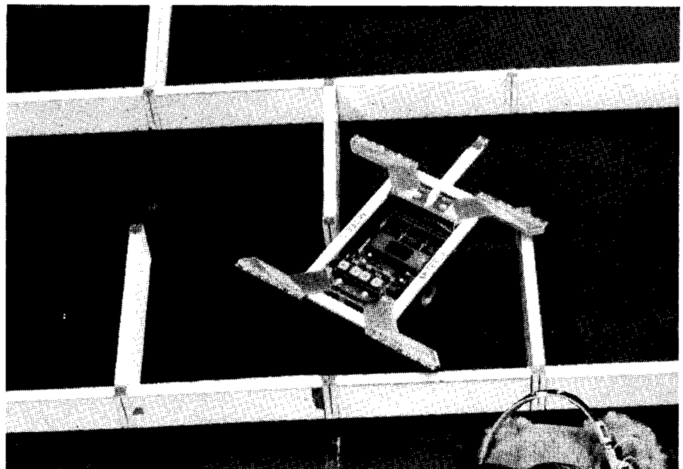
Here's to the Winners

The votes have been counted, the dust cleared, and the winners recognized in the First CIRCUIT CELLAR INK Design Contest. We received 26 entries and found each of them to show the kind of creativity and expertise we've come to expect in CIRCUIT CELLAR INK readers.

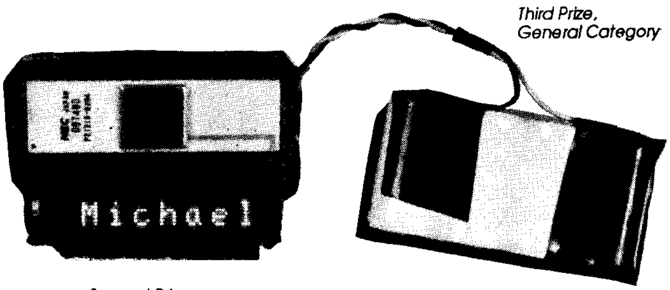
As you will recall, there were two categories in the contest:



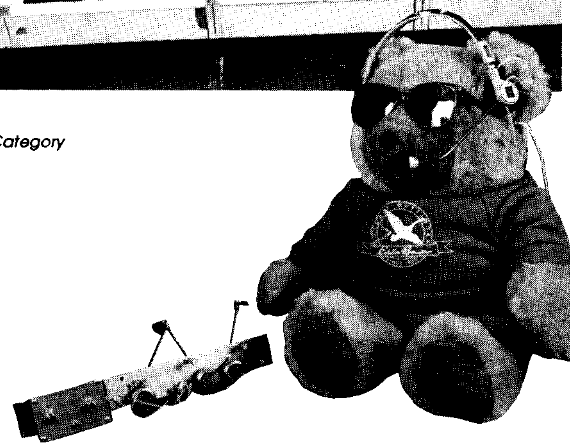
Third Prize, General Category



First Prize, General Category



Second Prize, Cost-Effective Category



Third Prize, Cost-Effective Category

First Prize, General Category-Mitee Mouse Three

David Otten, Newton MA

The Mitee Mouse Three is a self-contained robot designed to first navigate and map a maze, then traverse the maze in the shortest possible time. The judges were impressed by the skill of the design and execution, and by the advanced capabilities (including the ability to directly negotiate diagonals) of the robot.

First Prize, Cost-Effective Category-Portable DRAM Tester

John M. Wet-troth, San Diego CA

This portable DRAM Tester, based on the Intel 87C51, caught the judges' eyes for its simple design, functionality, and practicality. The tester checks 64K, 256K, and 1M DRAMs for access speed, data integrity, and pin functionality.

Second Prize, General Category-A Personal Computer-controlled Pipe Organ

Norman C. Jarosik, Princeton NJ

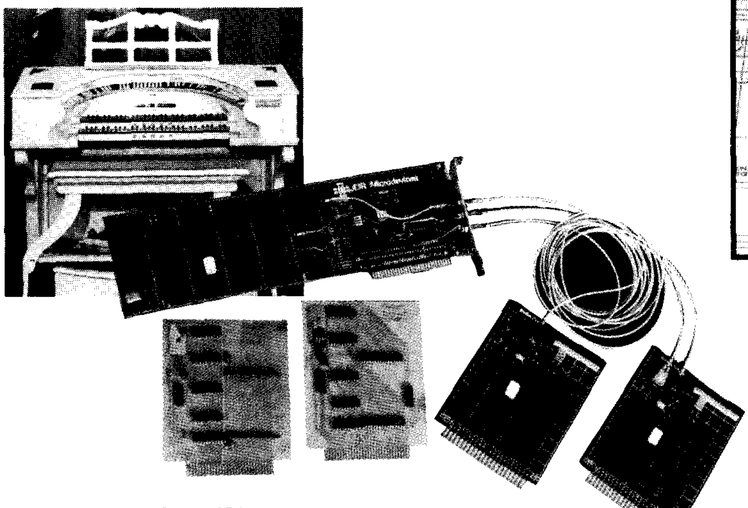
Take one standard theatrical pipe organ. Add an IBM PC-based controller (using three 8031 controllers) communicating with the organ over fiber-optic links. Shake with a liberal dose of software and you have the project with the highest **"gee-whiz" factor** in the contest.

Second Prize, Cost-Effective Category-An LED-Based Name Badge

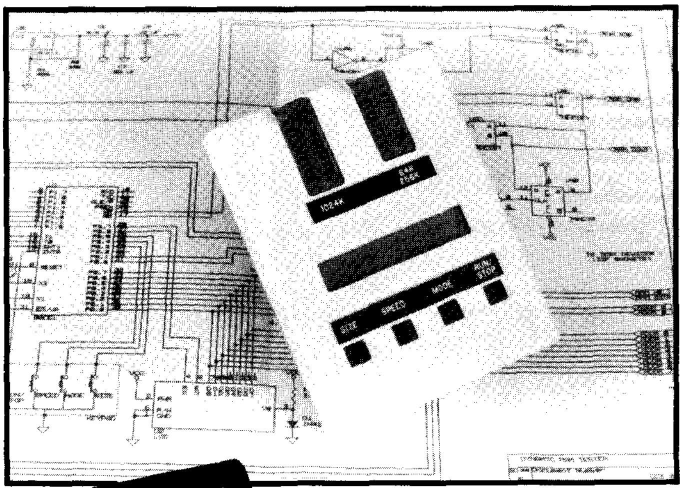
Michael L. Ardai, Allston MA

Michael Ardai used an 8748 to drive a Hewlett-Packard HDSP2111 5x7 dot matrix LED display and thus created a very snappy-looking electronic name badge. **The battery pack tends to make a breast pocket droop just a bit, but the creativity** shown **in addressing a simple problem was sufficient to garner second prize honors.**

General ***and*** **Cost** *Effective. We instituted the two categories* ***because we didn't feel that simple 8031 designs could be reasonably judged against desktop Cray wanna-be's. As it turned out, there were far more of the former than the latter, but the division*** *came in* ***handy for*** *letting us recognize* more ***types of*** *designs and philosophies. And now, to the winners.. .*

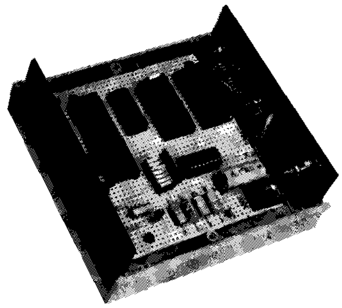


Second Prize, General Category



First Prize, Cost-Effective Category

Honorable Mention, Cost-Effective Category



Third Prize, General Category-8031 Analog Input Module

Rick Vaughn, Santa Fe TX

Analog input circuits are at the heart of most data acquisition and control systems. This design features an 80C31 microcontroller, a 10-bit A/D converter, and a 2-line x 16-character LCD to show readings for inputs, board address, and status and error messages.

Third Prize, Cost-Effective Category-Ruxpin 'Riter

Michael Eck, Martinsville NJ

If you have a child, you're familiar with Teddy Ruxpin, the robotic, story-telling bear. This project allows the user to make cassettes for T.R. These cassettes hold not only the audio portion of the story, but the motion control codes that operate T.R.'s mouth.

Honorable Mention, General Category-Digital Temperature-Indicating Controller (not pictured)

Francis Lyn, Rexdale Ontario

An 8031 microcontroller is the intelligence in this controller for baseboard electrical heaters. The design uses a resonant temperature transducer for input, and drives a high-current solid-state relay over a ten-second cycle as output.

Honorable Mention,
Cost-Effective Category—Infrared Printer Interface

John Runciman, Seattle WA

The HP82240A is a 24-character thermal printer which receives input via infrared signals. This 80C31-based interface converts standard RS-232 signals into the infrared required by the printer.

Congratulations to the *winners. You will* be seeing *complete details* of *most* of the *winners in* CIRCUIT CELLAR INK over the next few *months. We are also in the planning stages of a book containing the winning projects and worthy projects that we couldn't recognize here.*

We were so pleased by the response to this contest that we are planning the ***Second*** CIRCUIT CELLAR ***INK*** *Design Contest. Complete rules will be published in an upcoming issue, but it's not too early to get your pencil sharpened and your soldering iron warmed up for your next chance to show off your application design skills!*

Beyond ASICs

New-Generation *PLDs'* Cost *and Flexibility* Benefits

by Tom Cantrell

I'm sure you've seen the pundits' prophecy, "By 19*xx*, ASIC will make up *yy*% of commercial IC shipments." Just substitute some year for xx which is far enough away (usually 3-5 years) so no one will remember you said it, and plug in an outrageous number for yy-say 50–90%. There, now you're a "Guru" too.

"ASIC"—*Applica tion-Specific In tegrated Circuits. The* concept was that advanced IC tools and technologies such as gate arrays, standard cells, silicon compilers, and so on were to be given to system designers so they could design their own chips instead ofusingthoseboring"standard"chips.

1 say "was" because ASIC, like other over-hyped "concepts," continues to evolve in search of commercial viability. After figuring out that ASIC is a design and manufacturing technology, not a product, promoters have switched away from system designers and correctly targeted IC houses.

There are some successes--mainly bigbusinessand government customers. Lately though, "ASK" companies are introducing "standard" products as fast as they can. For example, VLSI Technology is making SRAMs and LSI Technology PC chip sets.

The original ASIC concept-the designer creates his own chip manufactured by an outside foundry-has largely failed for two reasons.

First, the tools aren't that great and designing chips still isn't easy. True, brainstorming your own architecture ideas is nice, but then you have to deal with messy realities like your trillion-cycle-simulated proto IC doesn't work and can't be tested. Proponents argue that the bright future for ASIC is still there right around the corner, as soon as "better" (smarter, cheaper, faster) tools are available.

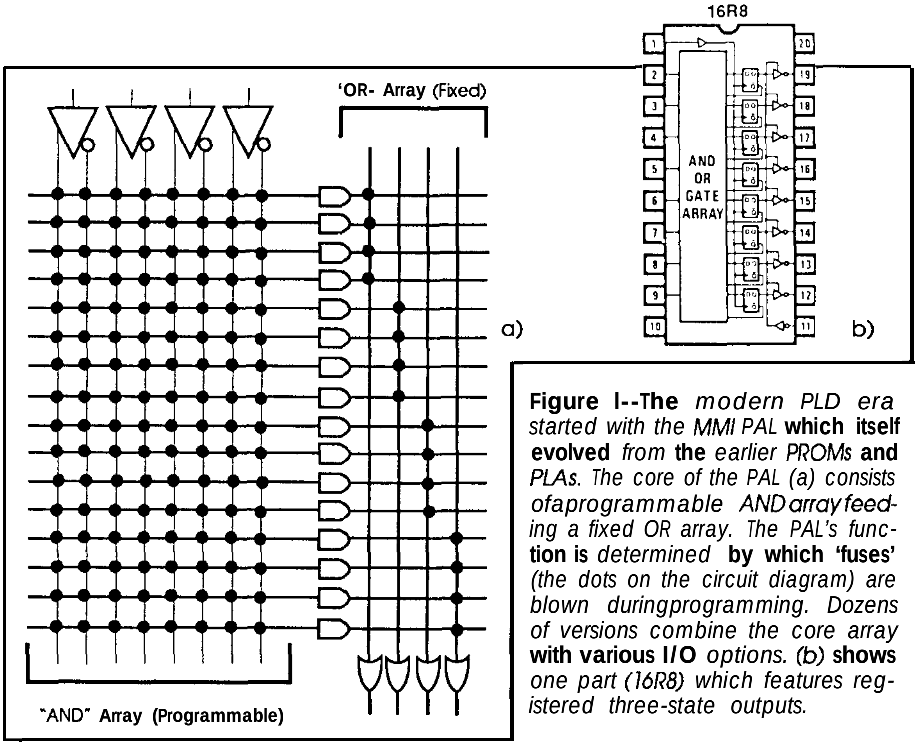


Figure I--The *modern PLD era started with the MMI PAL* **which itself evolved** *from* **the** *earlier PROMs* **and** *PLAs. The core of the PAL (a) consists ofaprogrammable ANDarrayfeeding a fixed OR array. The PAL's func*-**tion is** *determined* **by which 'fuses'** *(the dots on the circuit diagram) are blown duringprogramming. Dozens of versions combine the core array* **with various I/O** *options. (b)* **shows** *one part (16R8) which features registered three-state outputs.*

But the real showstopper for the original ASIC concept doesn't have to do with technology at all. Instead, it is the marketing "reality" that ASIC really means "custom" and all the risks associated therewith. For example.. .

Design Risk-Make sure you budget time and money for the four or five mask revisions it will take to get a chip you can live with.

Inventory Risk-Your company's ASIC-based "Talky Dick" politician doll went nowhere. Don't worry: The 100,OOOchipsinyourwarehouseought to make good tie tacks or fishing lures.

Sole-Source Risk-The place making your chips burns down. Switching your design to another foundry may not be as quick and easy as you think. MaybeN.R.E. ('Non-Recurring Engineering") really means NeveR-ending Expense.

PLDs TO THE RESCUE

Enter the PLDs—*Programmable Logic Devices.* Marketing-wise, the beauty of a field-programmable device is that it is a "standard" chip and thus eliminates the risks associated with "custom" ASICs. Furthermore, low design and device cost make the technology accessible to small- and medium-size companies.

PLDs are kind of like microprocessors except the programming language is hardware oriented-boolean equations, schematics, state-machine diagram-instead of instructions.

The **modem** PLD era started a decade ago with the introduction of the PAL *(Programmable Array Logic)* chip by Monolithic Memories Inc.

The PAL evolved from earlier generation PLDs—PROMs *(Programmable Read-Only Memories)* and PLAs *(Programmable Logic Arrays),* combin-

GAL20V8A	GAL1 6V8A
20L8	16L8
20H8	16H8
20R8	16R8
20R6	16R6
20R4	16R4
20P8	16P8
20RP8	16RP8
20RP6	16RP6
20RP4	6RP4
14L8	10L8
16L6	12L6
18L4	14L4
20L2	16L2
14H8	10H8
16H6	12H6
18H4	12H4
20H2	16H2
14P8	10P8
16P6	12P6
18P4	14P4
20P2	16P2

a)

PEEL18CV8	PEEL20CG10	22CV1 0(Z)	PEEL1531253	PEEL1 731273
10H8	12L10	12L10	PLS153 (82S153)	PLS173 (82S173)
12H6	14L4	14L4	16C1	12L10
114H4	16L6	16L6		20L8
16H2	18L4	18L4		20L10
16L8	20L2	20L2		
16R4	20L10	20L10		
16R6	20L8	20L8		
16R8	20R4	20R4		
10L8	20R6	20R6		
12L6	20R8	20R8		
14L4	20V8	20V8		
16L2	20G10	20G10		
16P8		22V10		
16RP4				
16RP6				
16RP8				
16V8				
16HD8				
16LD8				
18P8				
EP310				
EP320				
5C031				
5C032				

b)

Figure 2-h *the face of PAL-device-type proliferation, one trend-exploited by the (a)Lattice GAL (Generic Array Logic) and (b)International CMOS Technology PEEL (Programmabie Electrically Erasable Logic)-is to include smart programmable I/O logic allowing a single device to emulate many eariier ones. Both of these chips feature electrical erasability (thanks to EEPROM) as well.*

ing the best features of each (Figure 1). Using "fuse" technology, PALs were typically used by designers to consolidate glue logic-that miscellaneous TTL that inexorably creeps into every board design. PALs also served as "building blocks" in large systems. For an example, read how PALs saved the day at Data General in "The Soul Of A New Machine" by Tracy Kidder.

PALs languished at the start. MMI was a relatively small company pitching a novel concept to a skeptical audience of small-system designers. The first PAL chips themselves were relatively low density, slow, power hungry, expensive, and sole-sourced to boot. Meanwhile the original tools were on the order of truth-table and "minimize-it-yourself" boolean equation entry. However, PALs gradually started making inroads into a wide variety of designs. In the past few years, the PAL market has exploded.

Ultimately, the PAL became more than the company could handle, and MMI was acquired by AMD. But by that time, the original MMI PAL patent license had been seeded to a wide-variety of large and small semiconductor companies. Incidentally, I expect the original MMI (now AMD) PAL patent will be the subject of much litigation until it runs out—and probably after that too!

The real excitement now is that next-generation PLD architectures are emerging featuring advanced processes and ever more powerful and easy-to-use tools. Let's take a look at three new contenders.

NEW-AGE PLDs, LCAs, AND FPGAs

The first evolutionary track improves various aspects of the original PAL with upgraded architecture and process. Typical features of these "second-generation" chips include:

Speed/Power-One reason MMI is no longer with us is that they stuck with their old bipolar process long after everyone else had moved on. Most new contenders use high-speed, low-power CMOS processes.

I/O Programmability--The original PAL lineup included dozens of parts, each offering a particular variation of number of inputs and outputs, output clocking and polarity, and so on. By making the chip a little "smarter," manufacturers have come up with single devices that can be programmed to mimic many of the old versions (Figure 2). Obviously, dealing with just one part is much better for both supplier and customer.

Erusabilify-Instead of the permanent "fuses" of the original PAL, some suppliers offer EPROM- (Altera) or even EEPROM-based (Lattice) devices. Besides the obvious financial advantages for lab work, field upgrades, and so on, erasability makes testing (by the manufacturer and customer) easier.

Application Specificity-A generic device is "tuned" for a certain niche application by the addition of a small amount of dedicated logic. Examples are devices that include a microprocessor bus interface or are optimized for state-machines.

The list of first- and second-generation PAL/PLD manufacturers in-

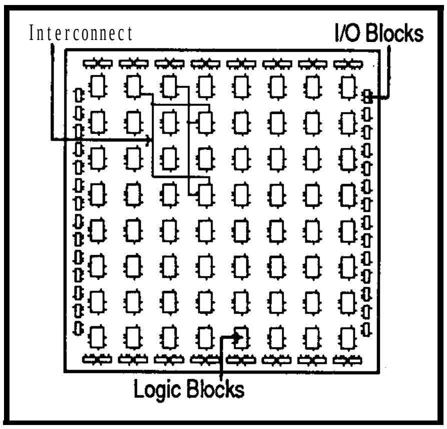


Figure 3—*The Xilinx LCA (Logic Cell Array)—the first truly new PLD architecture since the PAL-features high-function logic and i/O 'blocks' which are programmabiy interconnected to determine the chip's function. Unique among PLDs, the Xilinx parts are RAM based.*

cludes almost all semiconductor companies, with the notable absence of the top Japanese suppliers.

The first brand new beyond-PAL chip, the Xilinx LCA (*Logic Cell Army)* features two major innovations. First, thearchitecture,insteadofbeingbased on the PAL's array of AND/OR gates, utilizes an array of higher-functionality logic cells and I/O blocks (Figure 3). Second, the programmability is in the form of RAM, so it is not only "erasable" like EEPROM, but is "dynamically reprogrammable."

Upping the ante from AND/OR gates to logic cells (called CLBs for Configurable Logic Blocks) and I/O blocks (IOBs) allows an LCA to handle surprisingly complex logic: from 1200 equivalent gates (about the same as a PAL) to 9000 (about the same as a midrange gate array). These correspond to combinations of 64 CLBs/58 IOBs all the way to 320 CLBs/144 IOBs. A single CLB can perform combinatorial and clocked logic operations with multiple inputs and outputs. The IOBs allow inputs to be latched or direct, and outputs to be three-stated.

However, overall LCA performance depends on the cleverness of the routing scheme that connects signals between the cells; each "stage" of signal switching introduces delays. Xilinx uses a 3-level hierarchy of on-chip buses—*adjacent, general purpose,* and *long* lines-to optimize the tradeoff between speed and flexibility. If your application is relatively simple and/or low performance, you won't have to worry about routing. Otherwise, you must "tweak" your design to avoid critical path delays.

Meanwhile, the fact that the LCA is based on RAM is both problem and potential. The problem is the LCA RAM needs to be "initialized" each time the system is powered on. Xilinx eases the pain as much as possible by offering various initialization schemes. For example, the LCA can override your pin definitions to implement a parallel bus at power-up which a host MPU can then use to access the LCA like any other peripheral chip. Alternatively, if an MPU isn't present, a standard EPROM can be used to boot the LCA. Multi-LCA initialization is handled with a "daisy-chain" mode in which initialization data moves along the chain from the first to the last LCA (Figure 4).

The potential of RAM-basing, underrated in my opinion, is "soft" hardware. Using one or more LCAs, you could design a system with functionality largely determined by code loaded at power-up. A neat by-product of such a design strategy is that hardware fixes might be as simple as shipping a new ROM or disk, instead of laying out a new circuit board!

A similar PLD architecture is the FPGA *(Field-Programmable* Gafe *Array)* by Actel. Like the LCA, the FPGA features logic and I/O modules which are programmably interconnected to determine the chip's overall function.

Thanks to the use of "antifuse" technology-the preprogrammed state is "open''-the FPGAs, though not erasable, are highly testable. Further, two pins can act as probes providing access to internal signals (Figure 5)—a real plus for debugging.

Using hundreds of thousands of antifuses, Actel features an automatic place-and-route scheme that is similar to that of a channeled gate array ASIC. The result is flexible interconnection and 85-95% gate utilization.

FPGAs are available in densities from 1200 to 6000 gates. Noting that a typical TI'LMSI-equivalent function—such as a decoder, counter, or shift register-only consumes about 50 gates gives you an idea of just how powerful FPGAs are.

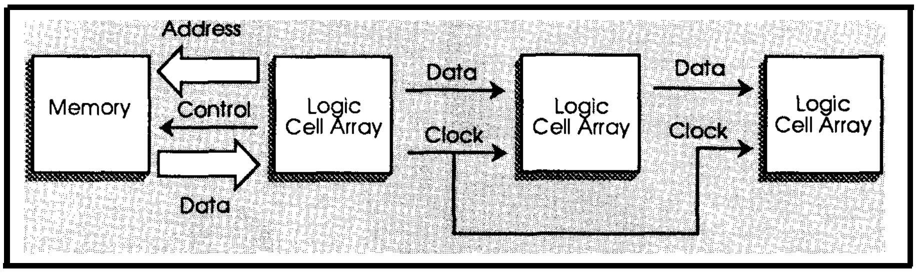


Figure 4— *Thanks to RAM, the Xilinx LCA can be dynamically reprogrammedbut also must be initialized at each power up. One of the many initialization modes offered uses a serial daisy-chain to load multiple LCAs while requiring a minimum of board traces and pins.*

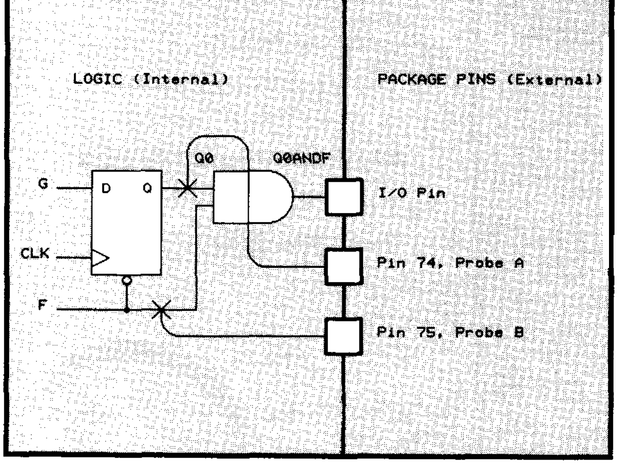


Figure 5—*Actionprobes are aunique feature of the Actel FPGA (Field-Programmable Gate Array) which allowreal-time in-circuit debugging by programmably connecting internal ('buried') signals to external pins.*

TOOLS ARE KEY

It's been said that programmable logic withou t good programming tools is useless. The good news is that tools continue to improve and PLD development is easier than ever.

Old and new PAL-type devices are now supported by popular third-party development tools running on PCs and workstations. Leading contenders, such as Data I/O's ABEL and Logical Devices' CUPL, allow you to enter, compile, and "bum" (program the PLD) your design. Add-ons bring schematic entry (supplementing the standard boolean equations), simulation, and automatic test vector generation to your bag of tricks. For a few thousand dollars, you can be the first on your block with a desktop "fab" which supports hundreds of types of low-cost PAL-type chips.

Meanwhile, the new-architecture PLDs—such as those from Xilinx and Actel-are also supported by manufacturer-supplied tools which fully exploit their particular technology. These packages feature schematic entry with "macro cell" libraries which implement common SSI/MSI functions, and include the programming hardware needed to put your ideas into silicon. ✣

Tom Cantrell holds a B.A. in economics and an M.B.A. from UCLA. He owns Microfuture Inc., and has been in Silicon Valleyfoy 10 years involved in chip, board, and systemdesign and marketing.

IRS

2 16 Very Useful
2 17 Moderately Useful
218 Not Useful

Actel Corp.
955 E. Arques Ave.
Sunnyvale, CA 94086
(408) 739-1010

Advanced Micro Devices
901 Thompson Pl.
Sunnyvale, CA 94088-3453
(800) 222-9323

Altera
3525 Monroe St.
P.O.B. 58163
Santa Clara, CA 95052-8163
(408) 984-2800

Data I/O
FutureNet Division
10525 Willows Rd. NE
Redmond, WA 98053
(206) 867-6988

International CMOS Technology
2125 Lundy Ave.
San Jose, CA 95131
(408) 434-0678

Lattice Semiconductor
5555 NE Moore Ct.
Hillsboro, OR 97124
(503) 681-0118

Logical Devices
1201 N.W. 65th Pl.
Ft. Lauderdale, FL 33309
(305) 974-0967

Xilinx
2069 Hamilton Ave.
San Jose, CA 95125
(408) 559-7778

FIRMWARE FURNACE

Exposing the Unseeable

Peering deep into code with the help of a single strategic bit

by Ed Nisley

You can identify an experienced engineer by noticing how many people are around for the first power-on of a new project. Old hands have far too many sea stories starting "When we fired up theol' Mumble Project.. ." to want any bystanders, innocent or otherwise, watching that first trial.

Quite often the hardware looks much as it did before the big switchflipped:perhapsafewLEDs glow,butlittleelsehappens. Without the sweet success of a functional system or the tell tale odor of electrical death, it is mighty tough to decide what the gnomes really are doing in there.

The topic this time is getting status and timing information out of your firmware, through one outputpinif that's all your system has available. I'll use code for both 8051 and PC systems to show you the similarities.

THE SIGHT OF ONE PIN FLIPPING

Most programs have short loops where the code waits for a specific event before continuing. That event can be as simple as a timer tick or as elaborate as a task scheduler searching for the next dispatchable task. Should the expected event not occur, the code jams.. .and the hardware locks up.

Lest you think only complex loops need monitoring, consider what happens when your code waits for the next timer tick after not enabling the timer interrupt. This sort of thing is remarkably easy to do and devilishly hard to find, particularlyifyoudon'tknowwhether the code is waiting for the timer or something else.

```
AVMAC51 assembler for 8051 processors

I I -
; Trace output routine
; A contains an ID code
; Bits go out MS3 first

ShowWait        PROC
                PUBLIC     ShowWait

;- send out 5-cycle-high indicator

                SETB       STALLBIT
                NOP
                NOP
                NOP
                NOP
                CLR        STALLBIT

;- send bits at 3 cycles each

                MOV        C,ACC.7
                MOV        STALLBIT,C
                MOV        C,ACC.6
                MOV        STALLBIT,C
                MOV        C,ACC.5
                MOV        STALLBIT,C
                MOV        C,ACC.4
                MOV        STALLBIT,C
                MOV        C,ACC.3
                MOV        STALLBIT,C
                MOV        C,ACC.2
                MOV        STALLBIT,C
                MOV        C,ACC.1
                MOV        STALLBIT,C
                MOV        C,ACC.0
                MOV        STALLBIT,C

;- send ending flag

                CLR        STALLBIT
                SETB       STALLBIT
                CLR        STALLBIT
                RET

ShowWait        ENDPROC
```

listing 1 —*ShowWait sends the value in the accumulator to a pin. The 805 1 subroutine adds approximately 38 µs to any loop into which it is inserted.*

If you assign each loop an ID number, then transmit that number each time around the loop, it is easy enough to decide which loop is in effect by watching the output values fly by. A single hardware output bit will suffice for this sort of identification if the code sends the ID number serially.

The `ShowWait` **subroutine in Listing 1 is an 8051 subroutine** that sends the accumulator value out a pin. The loop code must load the accumulator before calling `ShowWait`, so the whole affair adds about 38 µs to each iteration; in most cases this is entirely acceptable.

`ShowWait` **insertsdistinctive header and trailer markers around** each byte to set off the ID number between them. Because you will be monitoring the pin voltage on an oscilloscope, the code sends themost-significantbit first **to** put it on the left edge of the screen. Normal RS-232 serial data **isleast-**significant bit first.

Photo 1 shows what the ID number **17h** looks like on an oscilloscope. Because there are essentially no other loop delays, the header from one byte is jammed up against the trailer from the previous one. You can see why the headers and trailers must be of different durations!

Photo 2 and Listing 2 show a more realistic example taken from the **INKnet** 8052 firmware. The BASIC interpreter calls `Cons In` to get the next keystroke; `Cons In` **then calls** `ShowWait` **with ID byte 03h.** Because the delay between successive

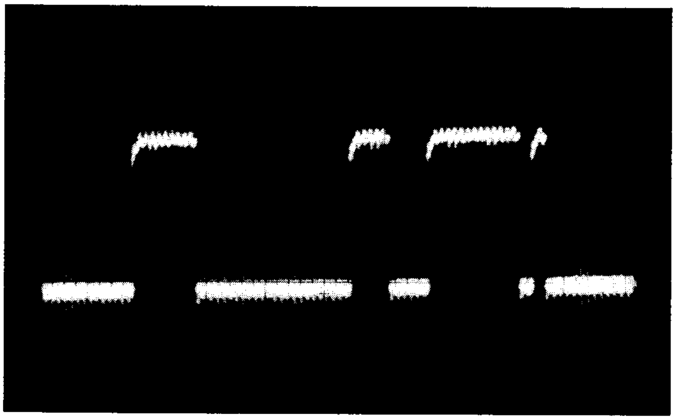


Photo 1—*When the header from one byte (in this case ID17h) is directly adjacent to the trailer of the preceding byte, it's crucial that headers and trailers be of different durations*

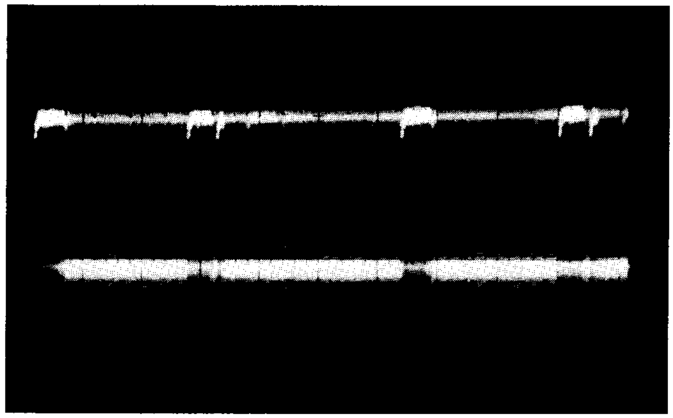


Photo *2-An oscilloscope screen showing the results of the code in Listing 2. The delay in the loop shows up as a 55-µs gap between trailer and header,*

`ShowWait` calls includes all of the code in Listing2 as well as part of the BASIC interpreter, there is about 55 µs between each trailer and the following header.

Also visible in Photo 2 are faint background pulses. These are due to interrupts from the 8052's Timer 1, which provides a 5-ms heartbeat pulse for the net firmware. Each Timer 1 interrupt needs about 20 µs, which is about two divisions at the scale shown in Photo 2.

Sending an ID number through a single pin is most useful for detecting "frozen" loops in the firmware and is less handy for tracking down long execution sequences. A logic analyzer hitched to the pin may be able to record enough information to decode a string of IDs, but the aggravation coefficient goes up rapidly after a while.

You do not have to send a complete byte if you monitor only a few routines. Perhaps four bits is a more reasonable number for most cases; this will reduce the time spent in the output routine by about a third.

Incidentally, should your project have very relaxed timing requirements, you can slow the individual bits down to eyeball speed and read them off an LED. I've only had to use this trick once, but it was a lifesaver!

```
AVMAC51 assembler for 8051 processors

;--------
; Console character input
; Gets next char from ring buffer
; Stalls if nothing is available

ConsIn    PROC
          PUBLIC    ConsIn

L?retest:
; anything there?
          CALL    ConsStat
          JC      L?gol

; indicate where we are
          MOV     A,#CONSEMPTY
          CALL    ShowWait
; and try again
          SJMP    L?retest

L?gol:

<<< remaining code omitted >>>

          RET

ConsIn    ENDPROC
```

listing *2-An example of ShowWait in use. ConsIn (a keystroke fetch routine) calls ShowWait with ID byte 03h. The total delay between trailer and header is about 55 µs.*

PRECISE SPEEDING

Once your code is running reliably, you can convert your telltale bit into a measurement tool to report when a routine gets control and how long it takes to finish. Instead of sending an ID number, your code simply turns on the output bit when it starts and turns it off when complete. Microcontrollers can do such bit manipulations in one instruction, so the tracing has almost no effect on the execution time of the remaining code.

Used in this manner, one output bit is restricted to monitoring a single routine. If your hardware has the luxury of a few more output bits, you can trace several routines at the same time. Otherwise, you will need to reassemble the code to choose which routine controls the single output bit.

For IBM PC firmware, a parallel printer port is a fruitful source of output bits. Most systems have at least one printer port and it is easy to add a second port dedicated to debugging.

Listing 3 shows the `SetTrace` assembly language macro for PCs. The `_TracePort` variable holds the printer port base address, which may be zero to disable tracing. The leading underscore in the variable name is mandated by the C code that initializes TracePort, so you may omit it if you are using another language. The AX and DX registers must be saved during the trace code, although you could dispense with this if you were careful. The remaining instructions read the existing printer port data from the output latch, set or clear the desired bit, and write the data back again.

Notice the difference between the tests on `_TracePort` and `SetBit`.

Circle No. 111 on Reader Service Card

CIRCUIT CELLAR *INK's*

ADVERTISER'S INDEX

IRS

INK Rating Service

How useful is this article?

At the end of each article and some features there are three 3-digit numbers by which you can rate the article or feature.

Please take the time to let us, at Circuit Cellar INK, know how you feel our material rates with you. Just circle the numbers on the attached card.

```
MASM assembler for IBM AT hardware

;
; Set or clear a trace output

SetTrace MACRO      TBit,SetBit
         LOCAL      noblip

         CMP        _TracePort,0  ; skip if no port
         JE         noblip

         PUSH       AX            ; save  bystanders
         PUSH       DX
         MOV        DX,_TracePort ; set up output port
         Punt
         IN         AL,DX         ; get current stuff
         Punt

         IF         SetBit
         OR         AL,TBit       ; set it
         ELSE
         AND        AL,NOT TBit   ; clear it
         ENDIF

         OUT        DX,AL         ; tell the world
         Punt
         POP        DX            ; restore bystanders
         POP        AX
noblip:
         ENDM
```

Listing 3—*SetTrace is an assembly language macro for IBM PC-type computers. If the value in _TracePort is 0, tracing is turned off.*

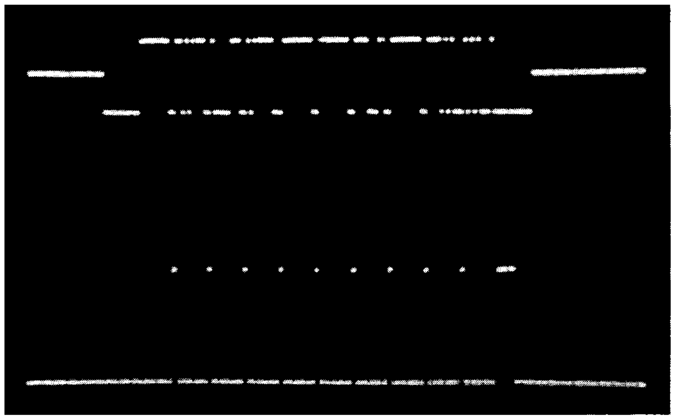


Photo 3—*The upper trace is data on INKnet measured at the RS-485 terminating resistor on the PC/AT network adapter. The lower trace is set by SetTrace called from the AT's serial interrupt handlers as it processes each byte.*

The CMP instruction operates while the program is running, while the IF statement selects an instruction at assembly time. The latter determines whether this invocation of `SetTrace` will set or clear the trace bit, while the former allows the **program** to dynamically enable or disable the entire tracing operation.

You must also be careful if you use the `SetTrace` macro in both interrupt routines and mainline code. It is possible for one instance of `SetTrace` to read the printer port, be interrupted by another instance that changes the actual port data, and then write the original (and now incorrect) data to the port. *You* must disable and enable interrupts around `SetTrace` if this could be a problem.

As an example of how this works, the upper trace in Photo **3** shows INKnet data measured at the RS-485 terminating resistor on the PC/AT adapter. The AT's serial interrupt handler uses `SetTrace` to set the bit in the lower trace while it processes each byte. The last message byte requires considerably more processing than the others.

```
MASM assembler for IBM AT hardware

;- interrupt status bits in output port
;  7 timer                     3 COM1
;  6 keyboard                  2 hard disk
;  5 interrupt 2 (video &c)  1 floppy disk
;  4 COM2                      0 DOS services (INT 21h)
; The HW interrupt bits must match the vector number
; The DOS software interrupt is special-cased

tracebits  RECORD  i08h:1,i09h:1,i0ah:1,i0bh:1, \
                   i0ch:1,i0dh:1,i0eh:1,dos:1


;--------
; Define generic hardware interrupt handler
; The int number must match the RECORD bit name
; Old interrupt vectors are stored in variables
;   OldIntxxh to match int number
; Presumes this will be assembled into COM file

HWInt    MACRO   intnum

         PUSH    AX
         PUSH    DX

         MOV     DX,CS:dataport   ; flag on
         IN      AL,DX
         OR      AL,MASK i&intnum
         Punt
         OUT     DX,AL

         PUSHF                    ; do old handler
         CALL    DWORD PTR CS:OldInt&intnum

         MOV     DX,CS:dataport   ; flag off
         IN      AL,DX
         AND     AL,NOT MASK i&intnum
         Punt
         OUT     DX,AL

         POP     DX
         POP     AX

         IRET
```

Listing 4-When **Tracer** is installed, the eight interrupt vectors from low memory are copied to the save *locations shown here. When the addresses of Tracer's interrupt handlers are inserted, any interrupt will invoke Tracer.*

The width of these pulses indicates that most characters require roughly 60 µs. I counted about 45 instructions between setting and clearing the bit, so the AT takes 1.3 µs per instruction. Averages like this can be dangerous, but for this stretch of code the AT is clocking 0.75 MIPS!

The INKnet code also speeds the AT's hardware Timer 0 up by a factor of 10 times the normal rate, so timer interrupts occur every 5.5 ms and the BIOS Time-of-Day clock interrupt is called every tenth timer tick. Set-Trace in the new timer interrupt handler showed 40 µs for the first nine interrupts followed by a 175-µs hit for the BIOS code link in the tenth interrupt.

PROBE PULSES

The first thing you learn about an oscilloscope is that it needs a trigger—a rising or a falling voltage-to start the beam across the screen. The second thing you'll learn is that a good trigger is hard to find.

Our lone output pin can serve yet another function: firmware can produce a scope sync pulse at any interesting point in the code. In fact, a few lines of firmware can produce an output pulse for the most unlikely combination of events; just the situation you need to monitor.

While you can buy some of this capability in a full-blown logic analyzer, a few lines of firmware are considerably cheaper and easier to control. Because the code can use "hidden" variables that don't appear on the processor bus, software can provide triggering capabilities far beyond those of any hardware analyzer.

The requisite code for ATs is similar to `SetTrace`, but it must set and clear the selected bit in quick succession. This will produce a two- or three-microsecond pulse (on an 8-MHz AT) that is entirely adequate for scope triggering. If your scope needs a longer pulse, it is easy enough to add a short delay between the set and clear instructions.

The INKnet program produces sync pulses just before sending a poll to node 0 and just before sending any data packets. For example, the trigger for Photo 3 came from the pulse occurring just before the node 0 poll. That pulse went into the external trigger input, while the two vertical channels show the bus data and interrupt monitor pin. The "A" sweep covered the poll and response messages. The "B" sweep and delay picked out just the response message from Node 0.

One reminder: you can't predict the future. There are some things firmware just cannot do, and one is generating a pulse just before an event that the code doesn't control. We would all like a "sync pulse" just before the next stock market crash, but (to the best of my knowledge) there is no firmware controlling that function.

PRACTICAL PULSES

Adding traces, timing blips, and sync points is easy if you control the source code. Sometimes, however, you need to measure existing software that can't be changed. On PCs, at least, this is reasonably simple as long as the event you want to measure uses either a hardware or software interrupt; write a TSR to hook the interrupt into your code, then proceed as above.

The downloadable files this month do just that. TRACER.COM is a 1K-byte TSR (Terminate-and-Stay-Resident) program that indicates how long your PC spends in each hardware interrupt, as well as for most DOS services accessed through INT 21h. *[Editor's Note: Software for this article is available for downloading from the Circuit Cellar BBS and on Software On Disk #10. For downloading and ordering information, see page 78.]* Admittedly, there are folks who aren't fascinated by this sort of thing, but if you've gotten this far.. .

You will need a printer port and an oscilloscope to take advantage of TRACER's capabilities. Without the port, TRACER can't produce the outputs. Without the scope, TRACER seems to do nothing except occupy 1K bytes of RAM! TRACER always searches for and uses the highest-numbered LPT port in your system, and will tell you which one it has selected. No harm will come to a printer attached to that port, but you cannot print and use TRACER on the same port at the same time.

The installation code (which is not shown here) copies the eight interrupt vectors from low memory to the save locations shown in Listing 4, then inserts the addresses of TRACER's interrupt handlers. From then on, any of the interrupts invoke TRACER first.

The handlers in TRACER turn on the appropriate output trace bit, then pass control to the old interrupt handler so it can operate normally. When the old handler returns control, TRACER turns the bit back off and returns to the original, interrupted program.

This process is shown in Listing 4, which is the text of a macro that generates the seven identical hardware interrupt handlers. The program line to create a handler looks like:

```
HWInt 08h
```

The macro appends 08h to the name of the handler, the old interrupt vector variable name, and the bit name in the trace port. Obviously, I had to assign all the names to make this work correctly.

Hardware interrupts automatically turn off the InterruptEnable bit in the CPU, so TRACER doesn't have to worry about being interrupted while setting the bit. Interrupts will also be off when the interrupt handler returns because the IRET at the end of the old handler restores TRACER's disabled flags. Sometimes things work out just right!

TRACER also monitors INT 21h, the DOS services interrupt. Unlike hardware interrupts, which may occur at any time, software interrupts are used to communicate between two cooperating pieces of code. TRACER must pass the contents of the CPU registers to the old interrupt handler without change, which means that the standard hardware interrupt service routine shown in Listing 4 just won't work.

For most DOS functions, TRACER proceeds just as before. However, a few functions either don't return or return with the registers in a peculiar state. Rather than worry about those cases, TRACER simply *removes* all evidence of itself from the stack and passes control directly to the old handler.

The registers must be returned unchanged to the interrupting routine, too. This includes the Flags register, which has such vital information as the Carry and Zero flags. TRACER must copy the Flags into the appropriate stack location so that when it performs an IRET, the CPU will restore the flags automatically.

Software interrupts also disable the hardware interrupt bit, so TRACER need not worry about interrupts while setting and clearing the DOS trace bit.

Watching the DOS bit on a scope is an education, because even the simplest program results in a blizzard of DOS activity. It is futile to measure "the time spent in DOS" for most programs, but it's fun to watch!

TRACER can monitor Only interrupt handlers installed before it; if you run a program that captures an interrupt vector and does not pass control along to TRACER, that output bit will remain low. As with most TSR programs, TRACER must be loaded last to keep control over its interrupts.

Some TSR programs actively steal their interrupt vectors back from any other program at every timer tick; TRACER will not fight back against such a TSR. You will get best results in a more-or-less vanilla system.

EXTRA-CREDIT PROJECTS

When you run TRACER you will see all the hardware trace bits pulse high when their interrupt is in effect. The DOS interrupt line, however, is high when the PC is sitting at the DOS command line waiting for keyboard input. This makes sense until you remember that the last program ended with a DOS call that TRACER filtered out, so the bit should be low. Right?

As an exercise, modify TRACER.ASM so that it outputs the DOS function number (stored in AH when TRACER gets control) through the printer port. That way you can see the last DOS call made to TRACER and figure out why the trace bit remains high.

Hint: DOS calls itself using both documented and (somewhat) undocumented INT 21h functions. Which call never returns?

You can also trace specific DOS functions on separate output bits. Each bit would then reflect the use of the calls you were interested in timing. This would reduce the blizzard of activity on TRACER's single DOS bit to something more manageable (and meaningful!) on several bits.

In any event, you should now be able to get your code running, then measure its performance to a gnat's whisker. In an upcoming column I will use these techniques to spelunk some instruction execution speeds. ..and perhaps gore a few sacred cows in the process. ✣

Ed Nisley is a member of the Circuit Cellar INK engineering staff and enjoys making gizmos do strange and wondrous things. He is, by turns, a beekeeper, bicyclist, Registered Professional Engineer, and amateur raconteur.

IRS

219 Very Useful
220 Moderately Useful
221 Not Useful

FROM THE BENCH

To Participate is to Win

by Jeff Bachiochi

I dislike contests in which there are winners and losers! Yes, competition often brings out the best in people, but we tend to give all the praise to the few who 'bring home the gold" and forget the balance. The CIRCUIT CELLAR INK Design Contest brought out the best in many of our readers. Every entry had something to contribute no matter how big or small. Hundreds of hours were spent on the concepts, designs, and construction of the projects entered. In addition, the judges have given many hours to each of the 26 entries submitted.

This month I would like to feature parts of a few entries. These entries did not place in the design contest, but that doesn't mean they are not winners! Every entry had a spark of genius.

PRINTER PORT COMMANDS PROPORTIONAL CONTROLLER

Often, a problem's simple solution will beg for construction whether it is designed specifically for a product or just internal use. Metal Analysis Inc. required total temperature control and logging of electric furnaces in the heat treating and strength testing of steel and other alloys. Mel Borchardt, a metallurgist for the company, determined that digital pulse-width-modulated solid-state relays would provide 0–100% control of the furnaces. He designed a proportional controller linking the PC used for control and data loggers from all furnaces, and wrote a simple command language to interface the PC to the 6502-based furnace controller. The controller interprets commands from the PC and regulates the power to each furnace by adjusting the power-on duty cycle for each.

The PC's parallel port, normally not associated with controlling anything more than a printer, is the communications channel between computer and controller. In this case, a five-byte command is sent via the printer port to the controller. The first four bytes are ASCII characters followed by a Carriage Return (fifth byte). The first byte (@, A, B, C, D, E, F, or G) designates which furnace (0, 1, 2, 3, 4, 5, 6, or 7) will be affected by the following three duty cycle characters. The duty cycle characters (000–999) indicate the percentage of on time (e.g., OOO=OO.O% on, 528=52.8% on, and 999=100.0% on). The controller uses the printer port error line to signal the PC that an invalid code has been received. The PC can clear the error by transmitting a "?" to the controller. Since the commands are all printable characters, a printed audit trail is easily read without any translation. This low-parts-count design including unique use of a PC's printer port equals success in any language.

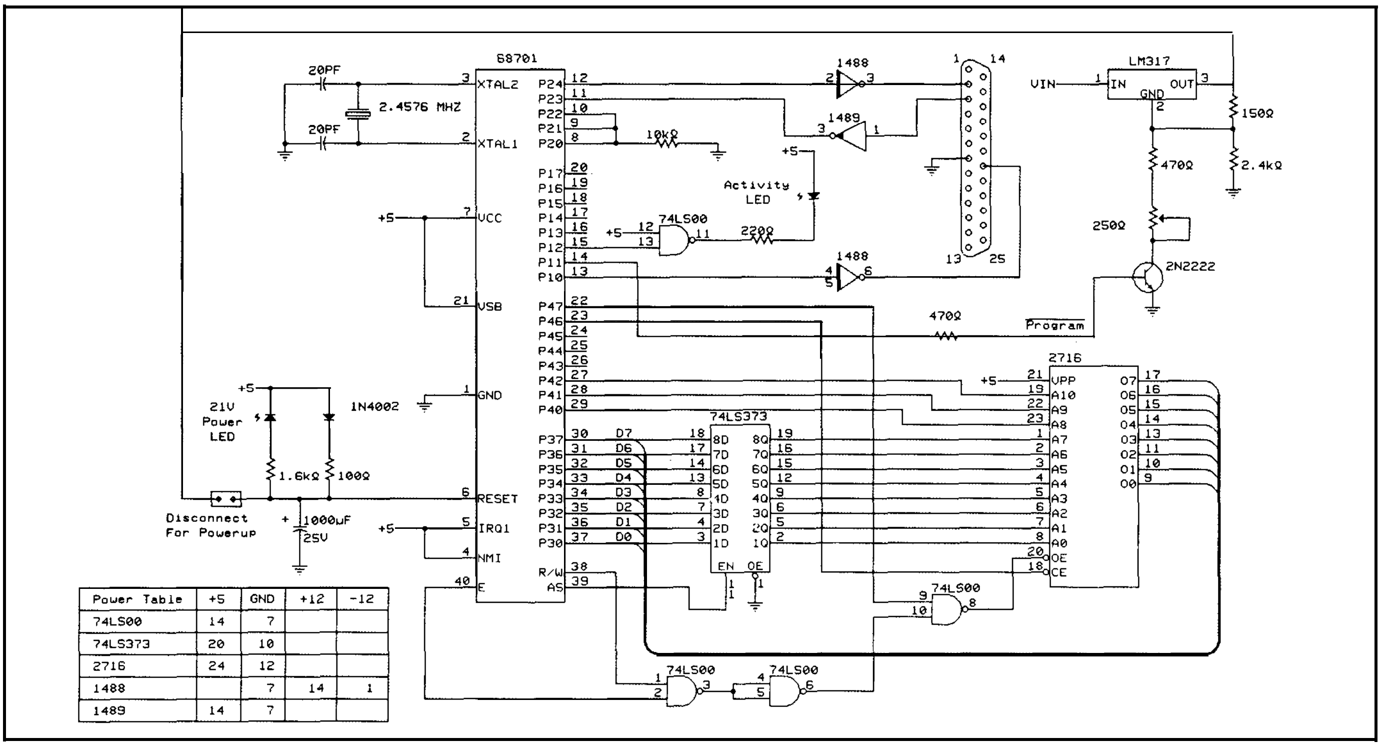
MICROCOMPUTER/CONTROLLER PROGRAMS SELF

Most programmable micros require a specially designed programming module to be programmed by a standard EPROM programmer. This expensive approach was avoided by Lawrence E. Wickliffe in his 68701 programmer design. External code, hand assembled by Lawrence and run at power-up, waits for serial transmissions from a host computer. Four-byte packets follow the same configuration as in the "Simple Serial EPROM Programmer" article by Steve Ciarcia in the February 1985 issue of BYTE. A high address byte (containing a read/write flag as the most-significant bit), a low address byte, a data byte, and a dummy byte acquired through the serial port of the 68701 initiatesa DTR\ (data terminal *not* ready) to halt communication. The appropriate address is programmed within the micro and DTR is again flopped signaling "ready for another packet." This loop is repeated for each address to be programmed.

The software used in the original article can be used with two minor modifications. The first is that the 68701's erased state is 00 hex and not FF hex as in EPROMs. The other is that the actual addresses do not start at 0000 hex but in the upper end of the 64K space within the programmable micro (Figure 1). Five chips, not counting the power supply, support the programming of the 68701: 1488 and 1489 level shifters, an address latch, an EPROM holding the external code, and one 74LSO0 for glue. Combined, they create an inexpensive circuit for experimenting with programmable micros.

LAYOUT DESIGN ELIMINATES NEED FOR DOUBLE-SIDED BOARD

Lower costs can be the result of many design changes. The parts used, quantity of the parts bought, and even the trade-offs of performing a function in software or hard-



Power Table	+5	GND	+12	-12
74LS00	14	7		
74LS373	20	10		
2716	24	12		
1488		7	14	1
1489	14	7		

Figure *I -A dedicated programming device isn 't necessary when the processing power of the 6870 1 is used to program itself.*

ware can affect the cost of a project. Tom Williams' tricks saved the price of a double-sided printed circuit board.

Tom chose to use the CDP1802 microprocessor in his design of a telephone security system used to monitor and log all incoming and outgoing calls. The 1802 was the first CMOS micro ever marketed! Customarily, interconnections between the micro and its memory cause a rat's nest of considerable proportions. By paying attention to con-

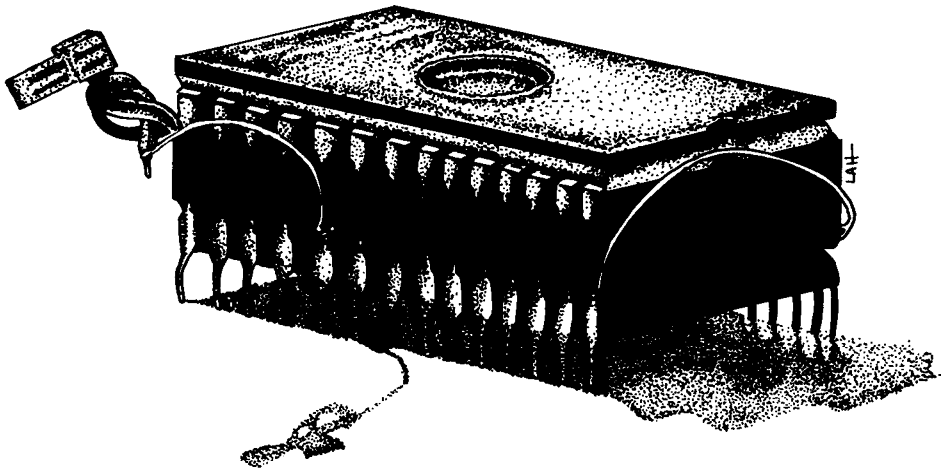


Illustration 1—*One of the easiest ways to conserve board space in one-of-a-kind projects is to piggy-back static RAM and EPROM.*

trol and higher address lines first, the subsequent address lines are connected to whichever address pins make the layout flow well as a single-sided layout design. A program doesn't care which actual cells are used, so the scrambling of address lines, or data for that matter, is unimportant. When it comes to external EPROMs, however, it makes a lot of difference. In order to overcome this, Tom prescrambled the data within the EPROM to match the addressing of the circuit layout. This also has the side effect of protecting one's code from disassembly.

In order to save additional board space, the RAM and EPROM were piggy-backed (Illustration 1). All but pins 1 and 20 on both devices have identical parallel pinouts. These two pins on the EPROM are wired separately to pads on the circuit board, with all others are soldered directly to the RAM, effectively reducing the circuit board size by two square inches. This method is not practical for production boards because the hand labor costs outweigh the circuit board cost savings. (I used this technique in the late 70s on my Radio Shack Model I to expand its memory from 16K to 32K bytes.)

MECHANICAL ADVANTAGE SPEAKS FOR ITSELF

Consumers have rejected most appliances which include vocal feedback. We don't like inanimate objects reminding us to change the oil (in our auto) or keep out of the refrigerator in between meals. Despite the resistance, digitized speech is becoming part of our culture. Ma Bell has been using digitized speech providing canned messages and information for years now. Keeping in step with this technology, Michael Eck designed a "personalized talking scale." Not only does the scale vocalize weight, it decides which member of the family it's weighing and gives personalized encouragement.

Open up a typical low-cost bathroom scale and you'll find a large disc that rotates when any weight on the scale counteracts the internal spring (Illustration 2). This disc is numbered incrementally and the appropriate weight spins into view through a small window atop the scale. This mechanical apparatus takes the place of a "load cell" which produces an electrical output proportional to the weight applied. (Load cells are generally expensive and used mostly in industrial applications.) By placing an optical chopper wheel (clear disc with black spokes applied) on the scale's rotating disc, two interrupter-type photodetector pairs can sense both direction and amount of rotation of the disc. A counter keeps track of the the spokes as they interrupt the light passed through the rotating clear disc. The number of spokes on the disc determines the resolution of the scale (number of spokes per pound). The biggest advantage of using this system is it becomes autozeroing, which prevents cheating. Whether by mistake or preplanned, the user can't misadjust the

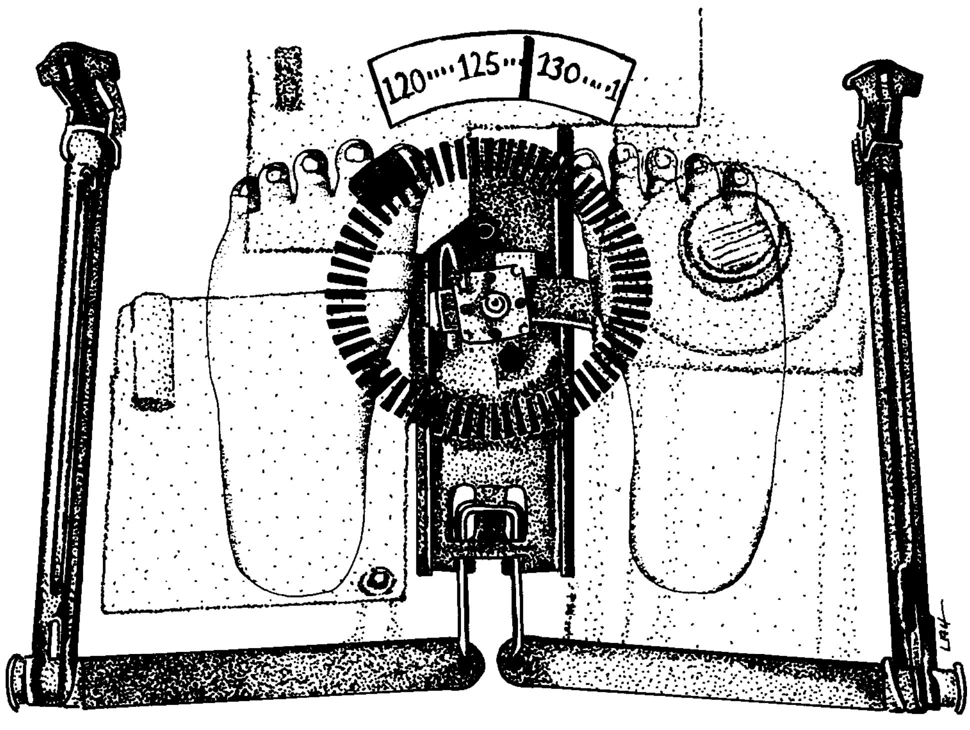


Illustration 2—*Most bathroom scales use a disc that rotates when a weight is placed on the scale.*

"zero" giving a false weight. The optical counter will make calculations upon the difference between no load and full load independent of the "zero pounds position."

AND THERE'S MORE...

Michael actually submitted more than one design. A second design creates a Microsoft Mouse-compatible bit stream from any joystick or trackball-type input device. The unusual part of this design uses an analog amplifier, a TL082, as a TTL-to-RS-232 level shifter (Figure 2). The

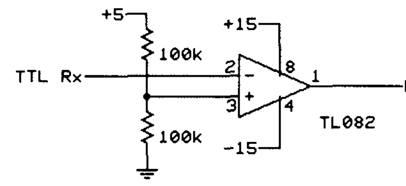


Figure 2—Analog *amplifier converts TTL to* **RS-232 levels.**

'82 has a maximum supply voltage of ±18 volts. It is short-circuit protected and has a slew rate of 13 V/μs. By biasing the positive input to 2.5 volts, TTL swings on the negative input will cause the inverted output to swing to within a few volts of the power supply. Two amplifiers are packaged within a 8-pin DIP; not bad for compact designs.

My hat is off to all who entered this year's CIRCUIT CELLAR INK Design Contest. It gives me a good feeling to know that hardware design is alive and flourishing in the old U. S. of A. Be aware that this does not mark the end of the contest, but the beginning of an annual event; an event where "To Participate is to Win." ❖

Jeff Bachiochi (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.

Innovations like these help to make today's technology more **cost efective,** reliable, and easier to use. Please share your favorite ideas, chips, circuits with others.

We will pay $25 for any From the Bench accepted for publication. All submissions should **be** typed, double-spaced, and include neatly drawn schematics or Schema configuration, library, and page files.

Include a stamped, self-addressed envelope large enough to hold everything if you wish the materials that have not been accepted to be returned.

Submit to:

From the Bench
c/o Circuit Cellar INK
P.O. Box 772
Vernon, CT 06066

IRS

222 Very Useful
223 Moderately Useful
224 Not Useful

SOFTWARE by DESIGN

Signal Smoothing

Taking the Rough Edges off of Real-World Data

by Jack Ganssle

Measuring low-level signals requires either very quiet (noise-free) electronics or clever signal averaging to smooth out the inevitable perturbations. Even if the electronics were perfectly quiet, quantization errors and resolution will always induce undesired signal characteristics that must be filtered out. Clever software algorithms are needed to extract meaning from what often appears to be little more than noise.

SIGNAL TYPES

Radio and TV receivers are astonishing devices. We digital people go nearly insane trying to remove a few millivolts of noise from our latest widget, while the radio plays a crystal-clear rendition of Mendelssohn's Octet. Often the station's signal is only a few microvolts at the antenna, buried in many millivolts of noise and rock 'n roll. Why must we fight so hard when digitizing nearly DC data, when a receiver works so well with three orders of magnitude less signal?

The music we hear is modulated on the station's carrier frequency. Marvelously designed RF filters can easily notch out virtually everything that is not at almost exactly the carrier's center frequency. The noise is distributed over a huge bandwidth; the filters remove all but a tiny portion of that bandwidth, including the noise.

Radios are a special case. Most signals presented to a digital system are at a much lower frequency. Indeed, most are nonperiodic, nearly DC voltages. With no carrier, all of the "bandwidth" carries important information.

The simplest example is the slowly varying output of a potentiometer (perhaps mounted at a robot's elbow joint). The signal changes slowly with time-for all intents it is DC. This is the hardest sort of signal to smooth.

Oscilloscopes display signals similar to those found in many digital instruments (indeed, many scopes are now entirely digital). The waveform is a swept signal, with one axis being time and the other voltage. Usually, each sweep is almost identical to the previous one; the waveform changes slowly in comparison to the sweep frequency.

Obviously, this sort of data can be filtered by averaging lots of successive sweeps. Perhaps not so obviously, the data can be smoothed on any one sweep along the implicit axis (the axis implied by array position-time for an oscilloscope, frequency for a spectrum analyzer or spectrophotometer).

This discussion is limited to data that is continuous along the implicit axis. This means there are no sharp discontinuities from one point to the next. In other words, for any point *i*, point *i+1* will be approximately the same value or will be generally changing in the same way the previous one was. Even a plot of the Dow Jones average versus time satisfies this condition, since the Dow does have a slight tendency to continue going up if it is on a rise, or to head down if its last change was bearish.

AVERAGING TECHNIQUES

A digital instrument reading simple DC-like voltages (as in the case of the robot arm pot) will most likely average multiple A/D readings in order to minimize the effect of noise. Little else is possible.

Of course, you must pick an intelligent averaging technique. Most programmers opt for a running average, also known as a boxcar. Only the latest *N* readings are included in the average, so very old data can't skew the results. *N* is chosen large enough to generate quiet data, but must be small compared to the data's rate of change.

For a running average, as each sample is collected the oldest is dropped and the new one added. A circular buffer is often used to store the readings. Once the buffer

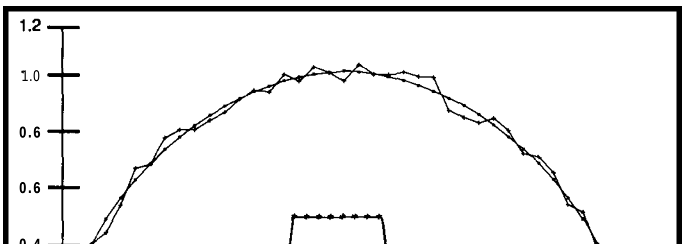


Figure 1 *—In a convolution, the input waveform is multiplied by a unit step function, and the results at each point are summed. The number of points in each average must be chosen to avoid both smearing the data and simply reproducing the input.*

is initially filled, the system can provide a smoothed output by taking only one reading and computing the average of the buffer's contents.

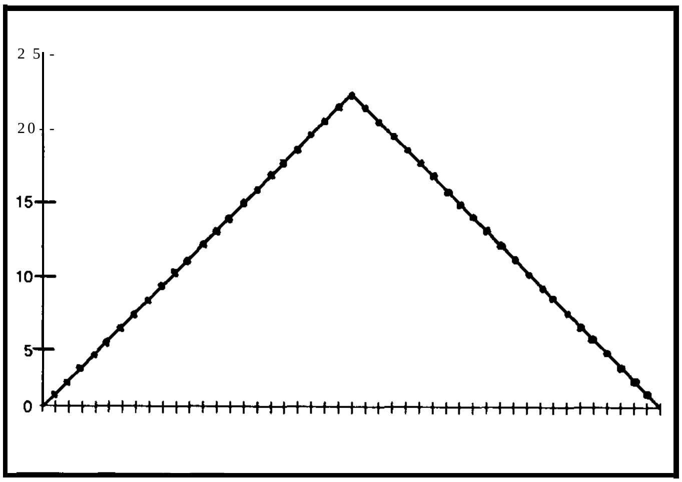


Figure 2—*When data is convolved with a function that resembl the input waveform, the midpoint is accentuated and succe sively outlying data deemphasized. Applied carefully, this pro duces a more faithful output waveform.*

Any simple averaging technique reduces noise at a rate proportional to the square root of the number of samples in the average. Using 100 samples results in an order of magnitude improvement in noise figure. Since any system will eventually be bound by the computer and A/D speed, this results in a sort of diminishing return. Increasing the number of averages to 1000 results in only a factor of three or so improvement in noise over an average of 100. There's a tradeoff between system response time and allowable noise level.

Asweepingsystemhastheoptionof averagingbothin time (by averaging each matching point of several sweeps together) as well as in the dimension the data is taken over. Since the data behaves as a **continuous** function, there are no sharp discontinuities between data points. Any point *i* is much like its neighbors, so averaging over the implicit axis doesn't distort the result excessively.

Although the runningaverage can be applied to a sweeping instrument, some take so long to acquire a single scan that their only option is to average in the frequency dimension-there simply may not be time to acquire more sweeps for the sake of averaging.

To average in the baseline dimension, imagine the digitized data has been placed in an array *D[i],* where *i* runs from 0 (the first point of the current scan) to *k* (the last point of the current scan). You can easily generate **a smoothed version of this data by com**puting a new array *D1[i..k].* Every element *i* of *D1* is the average of *D[i-2], D[i-1], D[i], D[i+1], D[i+2].* This is exactly the same as the boxcar average, except it is applied over the implicit axis.

Of course, this has the effect of smearing the output data. Sharp peaks will be slightly flattened. Can we define an averaging strategy that will avoid this side effect?

THE CONVOLUTION

The system just described generates **an** output for each point *i* that is the average of all the points in the vicinity of *i*. In effect, for each *i we* make an output by multiplying the input waveform by a unit step function, and summing the results at each point. The unit step is then slid one point to the right and the algorithm repeated. Figure 1 is an illustration of this process. Adjusting the number of points included in each average simply changes the width of the step. Wider steps (i.e., a bigger *N*) give more noise-free data, but at the price of smearing it. Narrow steps give noisy signals that are more faithful to the input data. A step exactly one point wide reproduces the input signal.

The technique of iteratively replacing any point *i* with a function of the points around *i* is called convolution. Widely used in image processing edge-detection routines, its applications are myriad.

Instead of using the crude unit step as shown, why not "convolve" the input data with a function that resembles the signal itself? In other words, pick a set of weights that accentuate the midpoint and include reduced levels of the more distant points. Figure 2 graphically shows a convolution with a triangular shape. This involves multiplying each point in the vicinity of *i* with a weighting factor, and then averaging the results. Points far away from the center play a much less significant role in the result, giving **a** more faithful output waveform.

Points	25	21	17	13	9	5
-12	-253					
-11	-138					
-10	-33	-171				
-9						
-8	147 62	-76 9	-21			
-7		84				
-6	222 287	149	-6	-11		
-5			7			
-4	322 387	249 204	27 18	8	-21	
				16	14	
-2	422	284	34	21	39	-3
-1	447 462	309 324	39 42	24	54	12
0	467	329	43	25	59	
1					54	17
2	462 447	324 309	42 39	24 21	39	12 -3
3					14	
5	387 422	284 249	34 27	19	-21	
6	322 287	204 149	18	-11 0		
			7			
8	222	84	-6			
9	147	9	-21			
10	62	-76				
11	-138 -33	-171				
12	-253					
Norm	5175	3059	323	143	231	35

Figure *3-A convolution can be arranged to provide a least-squares fit to input data, The integers shown are sets that provide this fit corresponding to various step widths.*

Since the sweep's data is continuous, wecanusuallyrepresentitsshape over a small interval with a polynomial of sufficient degree. Even noncontinuoussignalsareoftenpiecewise continuous; given a small enough interval their shape will also resemble a polynomial.

Remember that the sweep's data is noisy. Even if the sampled signal were a perfectly smooth waveform, by the time the computer reads the data, the less-than-perfect analog circuits and quantization problems may have made it more a collection of scattered points that only vaguely represents the sweep. No one point will be assumed to be correct; the curve defined by a collection of points will yield data that is, on the average, closer to the truth.

Many curve fitting techniques exist. In most instances the goal is to reduce the "amount of error" in the data. A least-squares approach does just this. Least squares will find coefficients for a polynomial describing a curve that fits the data in such a fashion that the "sum-square" error is minimized (i.e., the square root of the sum of the squares of the error at each point is a minimum). The error at each point is the difference between what the fitted curve predicts, and what the actual data is.

As you might imagine, computing a least-squares polynomial at each point of a sweep is computationally expensive. At realistic data rates no micro could ever keep up with the load. Fortunately, the concept of convolutions can be extended to do the least squares automatically.

The derivation is far too complex to present here. Suffice to say that a set of integers can be defined that, when convolved with an input signal, gives an output waveform that is a least-squares fit to an "ideal" signal over a narrow range.

Thisisreallythebestpossible solution to noise reduction. The sampled data is "idealized," and each smoothed point is then picked off the ideal curve. There isnomagicwiththemethod;noisy data will not be transformed into a perfect representation of the input signal. However, a much better result can be obtained than with a unit step. Since we may not have to average multiple sweeps, data can be smoothed without acquiring extra sweeps.

Points	25	21	17	13	9	5
-12	30866					
	8602					
-110	-8525	84075				
-9		10032				
-7	-20982 -29236	-43284	748			
-6	-33754	-78176	-98			
	-35003	-96947	-643	1133		
-5		-101900	-930	-660		
-4	-33450 -29562	-95338	-1002	-1578	86	
-3		-79564	-902	-1796	-142	
-2	-23806 -16649	-56881	-673	-1489	-193	1
-1	-8558	-29592	-358	-832	-126	-8
	0	0	0	0	0	0
1	8558	29592	358	832	126	8
2	16649	56881	673	1489	193	-1
3		79504	902	1796	142	
4	23806 29562	95338	1002	1578	-86	
5				660		
6	3.3450 35003	[illegible]	[illegible]	-1133		
7		78176	98			
8	33754 29236	43284	-748			
9	20982	-10032				
11	8525	-84075				
12	-30866 -8602					
Norm	1776060	3634092	23256	24024	1188	12

Figure 4—*When the derivation of the least-squares function is used, the results provide both a least-squares fit anddifferentiateddata. Theintegersshown compute the first derivative.*

Figure 3 shows a number of sets of integers that define convolving functions that will yield least-squares fits to input data. Different convolution intervals are given. A polynomial fit will be made over the number of sample points shown. Thus, the set of five integers will force a polynomial that tries to fit five sample points.

Using the convolution integers is quite simple. If the set of five integers shown in Figure 3 is used, then a smoothed signal D1(i) is made by computing:

*D1(3)= [-3*D(1) + 12*D(2) + 17*D(3) + 12*D(4) + -3*D(5)]/35*
*D1(4)= [-3*D(2) + 12*D(3) + 17*D(4) + 12*D(5) + -3*D(6)]/35*
*D1(5)= [-3*D(3) + 12*D(4) + 17*D(5) + 12*D(6) + -3*D(7)]/35*
etc.

The sum of coefficients times input data must be divided by the "norm," or the sum of the coefficients.

If the data is very busy (i.e., over only a few sample points it undergoes a lot of maxima and minima), then a small set of integers should be picked. After all, the method attempts to fit a curve to a short segment of the input data; busy data is all but impossible to fit under any circumstances. Data that changes slowly can use the larger sets of integers, resulting in more smoothing.

DIFFERENTIATION

The convolution integral has an important property, namely:

$$f'(t)=g(t)*h'(t) \quad \text{and} \quad f'(t)=g'(t)*h(t)$$

where * represents the convolution process and the prime marks indicate the derivative.

This means if you need to smooth and differentiate a signal, you can convolve the signal with the derivative of the convolving function. You never need to explicitly differentiate the input signal, a task that can use too much computer power to be practical.

It is therefore possible to compute the derivative of a continuous signal by convolving it with the derivative of the unit impulse (a step one point wide). Unfortunately, the derivative of the unit impulse is a function which cannot be expressed in a "discrete" form (i.e., a set of integers usable by a computer).

Since there are many instances where it is important to compute the derivative of an input signal, it is worthwhile to pursue this a little further. Remember that the convolving process primarily smooths input data. If a useful smoothing function is picked, the derivative of that function can be convolved with theinput signal toboth smooth and take the input's derivative. A single convolution can do both.

Although calculus is rarely mentioned in the computer world, we often use it without thinking about it. A derivative can provide extremely important results. For example, in image processing the edge-detection algorithms all rely on the convolution to compute the derivative of brightness levels.

By computing the derivative of the least-squares function, a new set of integers can be found that will both least-squaressmoothanddifferentiate thedata. Asetofintegers that smoothsand computes the first derivative is shown in Figure4. These values areused in exactly the same manner as previously described.

The convolution principle can be generalized to any order of derivative, and to almost any number of sample points. ✣

Jack Ganssle is the president of Softaid, a vendor of microprocessor development tools. When not busy pushing electrons around, he sails up and down the East Coast on his 35-foot sloop.

IRS

225 Very Useful
226 Moderately Useful
227 Not Useful

CONNECTIME *Excerpts from the Circuit Cellar BBS*

THE CIRCUIT CELLAR BBS
300/1200/2400 bps
24 hours/7 days a week
(203) 87 1-1988 — 4 incoming lines
Vernon, Connecticut

Conducted by Ken Davidson

The message base of the Circuit Cellar BBS in now available on disk. *See, page 78 for details.*

We're a bit short on space this issue, so I'll dispense with the usual ramblings and get right into the messages.

In the last issue, a group of BBS users discussed the good and bad points of various techniques for building the circuitry of a project (with special emphasis on wire-wrapping). Often the only difference between an experimenter's project and a professionally manufactured product is in the final presentation. A good-looking front panel can do wonders for first (and lasting) impressions, and a number of techniques for creating such panels are discussed in this first thread.

Msg#:18819
From: KEN RATKEVICH To: ALL USERS

I work in the Electronics Shop for a pharmaceutical company. One of my functions is to design and build miscellaneous circuitry for our scientific staff. One of my personal guidelines in this area is to provide devices that have a professional appearance to them. In other words, I want my creations to look like quality, off-the-shelf equipment. However, I run into great difficulty with the labeling of the controls, connectors, and indicators since I haven't found an in-house alternative to dry transfers. Things get really tough when I have to make labels oneletter at a time. I am very interested in learning about any alternative in-house methods for labeling that give professional results.

Msg#:18824
From: BOB DAMATO To: KEN RATKEVICH

Kroy has a lettering system out which gives real professional results. I have one at work, and it's fantastic. I can't find the address right now, but if you're interested 1'11 find it for you.

Msg#:18837
From: FRANK HENRIQUEZ To: KEN RATKEVICH

My old boss insisted that a project that looked great but worked so-so was far better than a project that worked great but looked so-so.

The Kroy lettering machine works great, but the letters rub off. And since they are on a clear plastic tape, the tape shows when you paste them on a panel (even after you spray the panel with a clear plastic protective spray). There's another machine that cuts the letters out of plastic tape. You can use the letters directly, orusethetapebackingasa stencil. I'll look for the company name and number tomorrow.

The easiest way I've found is to lay the panel out in MacDraw or a similar Mac program and print it on clear adhesive paper on a Laser-writer. You then peel the backing off, and stick the sheet to the panel. This works great, as long as your panel is no more than 8.5" x 14" in size. You can also spray Clearspray on it to protect the lettering, but even without a protective coat, the lettering will last a long time. Other than the size limitation, the only problem I've encountered is with accidentally trapping dust and air bubbles.

Msg#:18848
From: STEVE CIARCIA To: KEN RATKEVICH

We often use a method similar to that described by Frank (the fully drawn adhesive-backed template. To get around the 8.5" x 14" limit, however we use a stat camera instead of a laser printer. That allows us 3' x 3'!

Msg#:18852
From: JEFF BACHIOCHI To: KEN RATKEVICH

If you have access to a plotter and a program like AutoCAD, try designing the label with AutoCAD and then plotting the label mirror image on a piece of matte mylar. The design will not rub off because it's on the inside of the label and the outside will have a nice finish. Spraying the inside with a contrasting color will give it a nice appearance. Plotting to a file will allow other programs to use the file and plot to a laser printer.

Msg#:18868
From: BOB DAMATO To: FRANK HENRIQUEZ

I think Jeff has the right Idea. I do most of my highly detailed faceplates on AutoCAD then plot it out on the plotter. So far I haven't had an E-size faceplate! You can then plot it out mirror imaged on "chartpak," an adhesive mylar, or actually, not mirrored image. This way you can have many colors as well. You then stick it on the faceplate, and if need be, you can clear spray it.

Msg#:19267
From: GREG BELL To: JEFF BACHIOCHI

I find that cases/labels/faceplates for projects are sometimes the most difficult aspect of a project. I would love to see an article dealing with some of the more mechanical aspects of a project.

One example of a neat idea I've heard of but not seriously used is this: For a case, use plexiglass. Paint the inside of the plexiglass with a flat black paint. Neat appearance. Where you need to have an LED show through or a display, you place paper or tape of the appropriate shape on the inside before painting. That way, the face of the case is nice and flat.

Msg#:19273
From: JEFF BACHIOCHI To: GREG BELL

Sounds like we have the makings of an interesting message thread here!

Often times it seems as though the outside is more time consuming than the inside. Everyone likes function, but function is sometimes ugly! A nice wrapper can hide that ugliness and turn it into a work of art. :-)

Msg#:19309
From: FRANK HENRIQUEZ To: JEFF BACHIOCHI

Yea! I always feel that the case takes a disproportionate amount of time compared to the circuit. I wish someone would come out with good-looking cases (not those ugly gray aluminum boxes, or PC cases) that are somewhat expandable (I always find that a case is either too big or too small for the PC board). I wish someone would come out with a paper or plastic sheet that can be run through a laser printer and generate letters that can be rubbed off easily.

Msg#:19813
From: PELLERVO KASKINEN To: KEN RATKEVICH

I noticed there were no references to one very handy panel making system in the answers, so here's what I know:

We have been using for years something we keep calling Scotchcal according to the original release by 3M Company, even though it now is made by Letraset and called Dynamark Photomask. The materials are available in coated aluminum, plastic, and in orange-coated film that can be used to reverse the image from positive to negative.

The processing is simple if you have a blueprint machine or other similar source of ultraviolet light. Make a taped pattern on clear acetate film in 1:1 size. You can also use the Kroy text maker mentioned in other messages or rub-on letters. Then just stack the photomask material of your choice (and color), the master, and another clear acetate film for protection and feed the combination through the exposure. Takes 20 to 60 seconds on typical blueprint machines. Get the photomask material out. The exposed part has been polymerized (hardened) by the UV light. Gently rub the surface with a special solvent and soft cotton pads and the unexposed coating comes off.

With this process, if you start with color-coated aluminum material, you end up having a negative image (i.e., the aluminum shines through as the text). If you want a positive, you need to use the orange film at first in the same fashion. Then use the prepared negative as the new master for a second exposure.

You can coat the resulting panel with some spray lacquer, but not all lacquers produce good results. Submitting the ready panel to additional baking may make it more durable and, in general, we have not had many problems.

The actual box panel should have all the holes drilled before you apply the overlay, because any drilling will damage the looks. Instead, you just use a sharp knife through the holes to carve out the overlay where necessary. The aluminum panels are quite good against twisting when you tighten nuts. The same does not apply to the plastic overlays, where you need to be extra careful. Also, the corners of the overlays should be rounded, because any sharp corner tends to work loose in time. About a half-inch radius is enough.

Hope this helps you and other people in need of occasional neat prototypes!

A major component in any robotics application is the main drive motor (or motors). Finding such motors can be a problem unless one uses some ingenuity, as we find out in the following thread.

Msg#:13384
From: EDWARD WOOD To: ALL USERS

Does anyone out there have a source for some rather large DC motors for use as main drive motors in a robot? I've looked through the American Design Components catalogue but can't see anything with enough torque to do the job even with gearing down high-RPM ones. Any ideas?

Msg#:13423
From: JIM NELSON To: EDWARD WOOD

Well you don't exactly give any details on the torque you need, but you might give B&B a call: 800/638-7808.

A company called Magnetic Technology makes a brushless DC motor used in CAT scanners. I happen to have the catalog here: Peak torque is 1650 lb-ft. The unit's outer diameter is about 45 inches, and the inner bore is 28 inches. The unit resembles a ring, and is driven with three amplifiers phased 120 degrees apart. 818/887-7700

Have you ever read "Direct Drive Robots" by Asada & Youcef-Toumi? It covers the use of torque motors and brushless torque motors in great detail. No gearing in sight in their robots!

Msg#:13433
From: JACK PERGAL To: EDWARD WOOD

There's a company called Herbach & Rademan in Philadelphia that sells a lot of surplus parts. They used to have a lot of DC motors available. Call them at 800/848-8001 or 215/426-1708.

Msg#:15970
From: JOHN G. NAPOLI To: EDWARD WOOD

I've come across several types of 12 VDC automobile electric window motors that nicely fill the bill for motors with "oomph." Generally, older American cars utilized "monster" motors. Newer foreign cars, especially German (Audi) and Italian (Ferrari) cars with cable-operated lifts, are smaller and lighter. These motors are easily mounted and PTOed.

Msg#:16102
From: EDWARD WOOD To: JOHN G. NAPOLI

Is it possible to get these motors fairly cheaply? I imagine that a wrecking yard may have some available.

Msg#:16777
From: JOHN G. NAPOLI To: EDWARD WOOD

Yes, it really is. The best type of wrecking yards to do business with for stuff like this are those that let you in to remove the motors yourself. That saves them the labor, and allows you to also remove the miscellaneous brackets, relays, bolts, and linkages that come in handy when you want to use the motors in your application. The junk yards that remove and catalog the parts themselves charge for the service. Expect to pay half of list for these pieces.

For the first case, offer something related to the weight of the scrap parts. A window motor could go for several dollars—never offer more than ten bucks for any one part. This is all found money for these types of junkyards, since this stuff isn't removed from the cars before they go to the crusher. So they'll accept small "donations." The worst thing to do is to "spoil" them with excessive prices! :-)

To gain entry to a junk yard, ask for something weird like "a gas tank that's 13.5" long with an offset to the left." Don't ask for a specific part for a specific car. Works every time. Good luck!

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-l 988. **Set** *your modem for 8 data bits, 1 stop bit, and either 300, 1200, or 2400 bps.*

IRS

228 **Very Useful**
229 **Moderately Useful**
230 **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 (August/September 1989) includes all public messages posted during May and June, 1989.

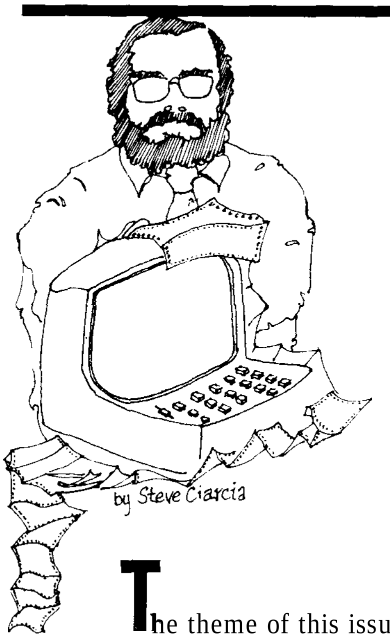
To order either Software on Disk or Circuit Cellar BBS on Disk send check or money order to:

Circuit Cellar INK — Software (or BBS) on Disk
P.O. Box 772, **Vernon, CT** 06666

or use your MasterCard or Visa and call (203) 875-2199. Be sure to specify the issue number of each disk you order.

Ctrl

STEVE'S OWN INK



Don't Mess With Mother Nature

The theme of this issue is among my favorite topics. Unfortunately, somebody may be trying to tell me that, like the Edsel, I'm a little ahead of my time. Let me explain.

For the last few months, I have been revising my home control system. The original HCS, designed and described in 1985, has been joined by a variety of new electronics including ROVER (see CIRCUIT CELLAR INK issue #5), a video-camera-based motion detection system, modulated lasers, and a multitude of distributed controllers. The original automatic lighting scheme is evolving into an AI environment where the "computer" can know who, what, where, when, and how much the inhabitants are doing. Of course, I haven't decided what to do with all this knowledge, but the sheer engineering challenge has been enough incentive to keep me stringing wires.

The central component of my latest home control adventure is a distributed control system which relies heavily on RTC52s (CIRCUIT CELLAR INK issue #8) and Ed Nisley's network software. During the past few months, I have been stringing low-voltage RS-485 cable and other signal wires among the house and two garages. This month I had hoped to be describing the control installation in an article, but Mother Nature intervened.

One of the continuing sagas of my life in the Circuit Cellar has been dealing with lightning. If you had told me that here in Connecticut, which averages barely 20 days per year of thunderstorms, that I'd have a lightning problem, I would have laughed. A hurricane or two every 50 years, but lightning?

Last week I was sitting on the deck sipping a cool drink. It was dark out and the floodlights had come on just as it started raining. There was even a bit of thunder, but it seemed far away and quite natural for a summer shower. Because I had installed a canopy over the greenhouse and deck, we could even sit outside in heavy rain without getting wet. As I tipped my glass to take another sip, I instinctively glanced toward my 15-foot satellite dish as a bolt of lightning went through the trees, around the lightning rod at the top of the dish, and hit directly on the LNA. BLAM! Open mouthed, I wiped the ice cubes from my lap and stared at the dish as a large cloud of steam rose from the area of the strike. All I could think was here we go, Blown Up Again!

After running around the house inspecting the damage I came to the conclusion that the hard-wired control system wasn't going to hack it any more. Even though signal connections between systems were optoisolated, this strike didn't go through the AC power system (now quite ruggedized) like all prior hits, it "jumped" across to the low-voltage control and signal lines. ZAP! Trash one house-full of electronics!

Rather than continue stringing more "lightning paths," I've decided to look harder at the connections between the remote units and the central HCS. Perhaps I have to go to fiber optics or build optoisolators with 200-kV isolation before I can use my new networked controllers.

The ultimate answer may be CEBus. My entire control system is predicated on the concept of closed-loop control and redundancy. Until now I have had no real commercial alternative, and the only option has been to think of the home as a process-control environment and adapt everything to it. Custom configurations are neither easily explained nor modified.

CEBus offers a common control medium where the designers of appliances and the designers of controllers can communicate intelligently. There is no need to stay in a hazardous "copper environment" when CEBus offers closed-loop control via the power line, IR, RF, coax, twisted pair, and fiber optics.

While these standards are still preliminary, there will be a solution for me here somewhere, and you can count on our continued interest and commitment to CEBus. In the meantime, quotes are coming in for installation of an industrial-quality lightning system for the house and garages to help direct the lightning along less troublesome paths. We're also working on an RS-485 isolator. Of course, to keep from sitting in the dark while a bunch of mute computer nodes mumble endlessly to themselves, I may have to install a couple of modulated laser beams between the buildings. At least it isn't too foggy up on this hill! Stay tuned.

Steve Ciarcia