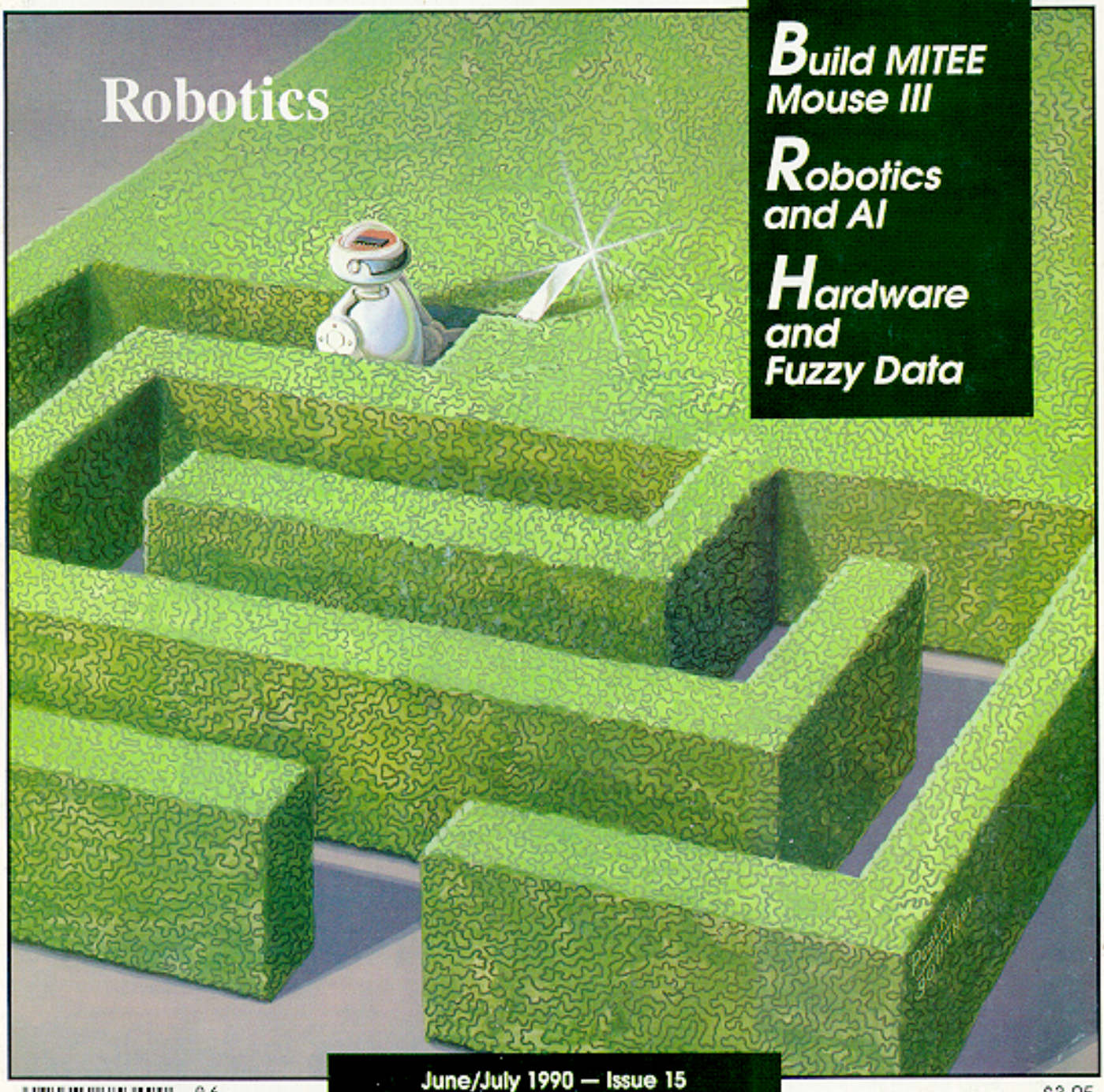
CIRCUIT CELLAR INK®

THE COMPUTER APPLICATIONS JOURNAL

BONUS SECTION

Intelligent Buildings



Build MITEE Mouse III

Robotics and AI

Hardware and Fuzzy Data

June/July 1990 — Issue 15

$3.95

06
0 74470 75349 0

Applications for All

EDITOR'S INK

Curtis Franklin, Jr.

I am writing this editorial two weeks before "Earth Day 1990." Articles have been written, television specials taped, songs sung, and marches planned. My local Big Chain Bookstore has a huge display of books, each with the word "Green" in the title. As a resident of The Earth, I think that taking care of it is a pretty fine idea. I enjoy planting (and sitting under) trees, and I've just about gotten the hang of this breathing business. I'm in agreement with much of what's going on with Earth Day 1990, but there is one large theme running through the middle of the whole shebang that worries me silly-the antitechnology theme.

The theme runs something like this: The tools used to muck up the environment involve technology, therefore all technology is bad. The logical conclusion is that, if we can just be rid of technology, everything will be As It Should Be. The definition of "technology" tends to be a little slippery in these discussions, with some folks holding to a nuclear fission" meaning, others joining in at the "from internal combustion to the present" point of view, and still others bringing up the vanguard with the "anything beyond a stone ax" position. From where I sit, it doesn't make any difference where you pin the definition because the basic theme is critically flawed.

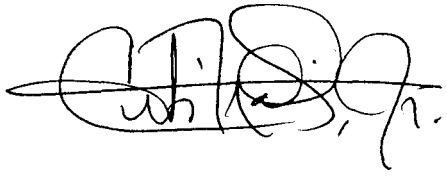
It's true that the speed of technological change has led us down some paths where we're not comfortable. Unlike working with horses or planting seeds, where humans had thousands of years to develop rules and limits of behavior, many of the technological advantages that we take for granted have been around for fewer than 150years. It takes time to ingrain rules in a society, and the time for building rules about technological change has been vanishingly short. None of this means that technology, and by extension the people who work with technology, are Evil. It does mean that it's important to apply critical thought to the implications of an application, but that's a process that's already begun.

There are many problems with antitechnology fervor as the basis for policy, and one of the worst is that it excludes many fine people from the process. CIRCUIT CELLAR INK readers are generally up to their elbows in technology. As engineers, programmers, and researchers you are accustomed to searching for workable answers. I'm certain that enlisting your help would be far more productive than damning the fruit of your labor. Unfortunately, the strains of "Down with Technology" seem unlikely to disappear from the lips of many in the Green Movement. Despite that, I'm confident that CIRCUIT CELLAR INK readers will continue to look for answers, think critically, and provide solutions to problems large and small, just as you've always done.

LOOKING INTO THE FUTURE

We've begun looking into the themes for CIRCUIT CELLAR INK in 1991 and I would like to get your help. If you are working in a particular area of applications, or would like to see more articles on a subject, send me a letter. I'm especially interested in hearing from readers who feel up towritinganarticlefora themeissue. Inaddition to themes arranged along the lines of the ones we've run so far, I'd like suggestions for "vertical" themes. Should we have an "Applications in Oceanography" theme? Perhaps "Computer Applications in Agriculture.. . Astronomy.. . Automobile Racing... Chemical Processing.. . Conservationand Ecology.. . or Animation"? I need to hear your ideas for particular themes.

One theme that I'm convinced we need to pursue stems from an editorial I wrote several months ago. In that editorial I told you about Joe Sobieski, a retired engineer and executive who spends much of his time working on cost-effective applications for the handicapped. I knew that there would be a response to the editorial, but I had no idea how many people would write. We have received more letters concerning Joe than for any other article or topic mentioned in the magazine. Researchers, educators, and engineers from around the world have sent mail. Now, I'd like to hear what everyone is doing in applications for the handicapped. If you have built an application (hardware, software, or both) to assist a physically challenged individual, please let me know. If you can write an article telling others how to duplicate or learn from your application, send me a letter. A theme issue on applications for the handicapped is one that I've dreamed about for quite a while, and I think that the readers of CIRCUIT CELLAR INK are just the folks to help pull it off.



FOUNDER/ EDITORIAL DIRECTOR
Steve Ciarcia

PUBLISHER
Daniel Rodrigues

EDITOR-in-CHIEF
Curtis Franklin, Jr.

PUBLISHING CONSULTANT
John Hayes

ENGINEERING STAFF
Ken Davidson
Jeff Bachiochi
Edward Nisley

CONTRIBUTING EDITOR
Thomas Cantrell

NEW **PRODUCTS EDITOR**
Harv Weiner

CONSULTING EDITORS
Mark Dahmke
Larry Loeb

CIRCULATION COORDINATOR
Rose Mansfield

CIRCULATION CONSULTANT
Gregory Spitzfaden

ART & PRODUCTION DIRECTOR
Tricia Dziedzinski

PRODUCTION ARTIST/ILLUSTRATOR
Lisa Ferry

BUSINESS MANAGER
Jeannette Walters

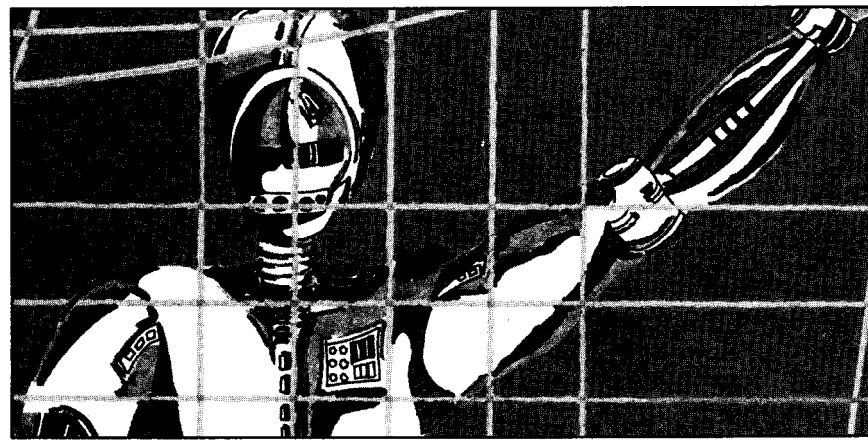
STAFF RESEARCHERS

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

Cover Illustration
by Robert Tinney

CIRCUIT CELLAR INK®
THE COMPUTER APPLICATIONS JOURNAL

In This Issue...



18 **Robotics and Artificial Intelligence**
Modeling Synthetic Actors and Real-World Interactions
by Chris Ciarcia

Robotics, animation, and artificial intelligence all come together at the point of building structures that respond to their environment. A look at the important similarities can bring new direction to your robotics work.

DEPARTMENTS

Editor's INK
Applications for All — **1**
by Curtis Franklin, Jr.

Reader's INK—*Letters* **to** *the Editor* — *5*

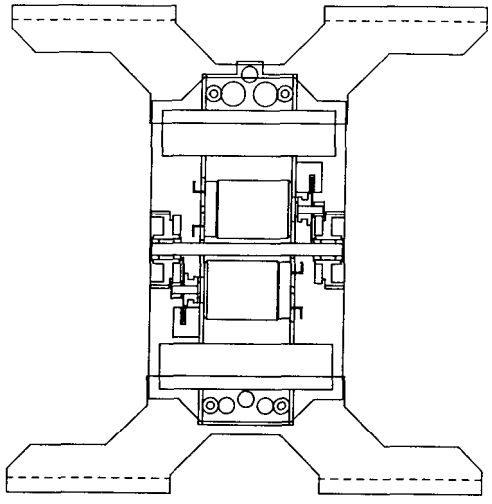
NEW Product News — **8**

Visible INK—*Letters* **to** *the INK Research Staff* — *12*

Firmware Furnace
The Furnace Firmware Project — 49
Process Control on the Home Front
by *Ed Nisley*

16 **Implementing a ComeFrom Statement**
Discover Where Your Code Has Been
by J. Conrad Hubert

Sure, everyone talks about a ComeFrom — but no one codes one. Jim Hubert decided to stop talking and start programming, and this useful technique is the result.



32 **Building MITEE Mouse III**
Part 1 — *The Hardware for a Maze-Running Rodent*
by David Otten

The Micromouse contest is a respected event in the world of robotics. Autonomous robotic mice must solve a maze, and maximize for time. This CIRCUIT CELLAR INK Design Contest Winner has won Micromouse competitions around the world. Part I shows the hardware for the robotic rodent.

41 **Building étude**
Part 2 — *A 25-MHz Analog-to-Digital Converter for the PC Bus*
by J. Conrad Hubert and Dick Hubert

Part I gave you the hardware, now learn about the driver routines for a cost-effective 25 MHz A/D converter board.

From the Bench
Power Control Basics — 57
Choosing the Best Digital Power Control Option for your Application
by Jeff Bachiochi

Advertiser's Index — **65**

Silicon Update
Chips for Artificial Intelligence — 68
I've Seen *the* Future-and *It Is* Fuzzy
by Tom Cantrell

ConnecTime—*Excerpts from the Circuit Cellar BBS* — *74*
Conducted by Ken Davidson

Steve's Own INK
Contemplation — 80
by Steve Ciarcia

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

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

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

CIRCUIT CELLAR INK (ISSN 0896-8985) is published bimonthly by Circuit Cellar Incorporated, 4 Park Street, Suite 20, Vernon, CT 06066 (203) 875-2751. Second-class postage paid at Vernon, CT and additional offices. One-year (6 issues) subscription rate U.S.A. and possessions $ 14.95. Canada/Mexico $17.95, all other countries $26.95-surface, $38.95-air. All subscription orders payable in U.S. funds only, via international postal money order or check drawn on U.S. bank. Direct subscription orders to Circuit Cellar INK, Subscriptions, P.O. Box 3050-C, Southeastern, PA 19398 or call (215) 630-1914.

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

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

Letters to the Editor

READER'S INK

MYSTERY CHIPS CLUE

In your April/May '90 issue (#14), there was a question in 'Visible INK" about some parts with the following markings:

i wp9083511
s70099
u6250248s

They are probably AT&T numbers. The number "wp9083511" is an in-house number for the equivalent of a 27512-25 64K x 8 NMOS UV EPROM. The "i" on the part is probably the Intel logo. The other two lines are probably IDs for the programming information.

Bill Carpenter
Freehold, NJ

DIGITAL SIGNAL PROCESSING

Dean McConnell's article on Digital Signal Processing (DSP) in CIRCUIT CELLAR INK #13 was a useful introduction to the topic. However, some precautions are necessary when applying the methods in the section "Replacing Analog Circuits," including Figure 8.

The "DSP Equivalent Software" operations for replacing nonlinear analog processing methods of rectifying, limiting, and so forth, create samples of nonbandlimited analog signals. These samples are, therefore, aliased, and are not representative of the analog signal's true frequency content. If further processing is dependent on the frequency content of the signal, such as in a DFT or digital filter, errors are likely.

Some nonlinear operations which create bandlimited signals, such as modulation, can be handled if the sample rate is either initially high enough, or if interpolation is used to increase the effective sample rate prior to the modulation.

Also, in his FIR filter sample, a filter length of 15 taps doesn't mean the output lags the input by 15 samples. For sinusoidal steady-state, a FIR filter of length N, with symmetric h(n), has a constant group delay (due to linear phase shift) of (N-1)/2 samples. The delay of seven samples in his case can be verified by a DFT of input and output (during steady state).

McConnell's first sentence, "...DSP...conjures up thoughts of exotic, complex mathematics and advanced electrical engineering theory," is very true. Intuitive approaches, while seemingly convincing, often cannot be justified mathematically.

Steven E. Reyer
Bayside, WI

OF 6800s AND DAAs

I know this is late, but I agree with Chuck Yerkes's letter in CIRCUIT CELLAR INK #11. I think it's important not to ignore the 6502 and 6800 family trees.

My interest in the 6502 arose when BYTE was two months old, when the magazine ran an article on the CPU. Knowing nothing about software, what attracted me to the 6502 was its cheap price. It took me about three years (you might say I grew up with BYTE), but when I did get a computer, it used the 6502. I've stayed with the 6502/6800 trees since then. Sure, I bought a Model 100 laptop, but don't program on it because the mnemonics look funny. And how much easier it would be to program if it used a 6809 with its relocatable machine language, instead of the 8085 which doesn't even have relative branches.

I also thought that it was important that [Mr. Yerkesl mentioned OS9. Supposedly it is quite popular in industry, especially as part of controllers, but the popular computing press has fairly ignored it. But I did see one article in a UNIX magazine about a year ago, which included it because they were discussing alternative operating systems for when UNIX isn't fast enough.

Having used OS9 as an operating system for applications in my Color Computer, I'm now starting to see its usefulness in controller applications. Instead of going the mainstream 80x86 route, with its ever-increasing demand for higher speeds and more **memory, it seems much** more valid to expand my system by offloading various features to their own little microprocessors. The features that make OS9 good, its relocatability, ROMability, small size, and ability to load correct versions of modules and ignore earlier versions will make this task easy. I haven't yet decided whether I'll leave OS9 in the "coprocessors" or replace it with simpler code once the main code has been finished.

I also want to make a comment about the DAAs also mentioned in #11. It seems like some of the cost of DAAs must be that they use old technology, mainly a transformer and discrete components. Those cheap phones get away with no transformers, so obviously the designers have overcome the problem of connecting solid-state devices to the telephone, where some of the voltages can get quite high. I don't see why some enterprising IC manufacturer doesn't design a solid-state DAA IC, and get it approved. Hopefully, it could be made cheaper than those discrete components used in various answering machines and modems, so the demand would be there. It might also allow those of us who need to build custom phone devices to legally, but cheaply, hook them up to the telephone lines.

I don't know if cheap computers have changed anything, but the deaf have Teletype systems for communicating over the telephone line. Since the design originated in the days of clunky and slow Baudot Teletype machines, the system is slow and thus the modems are very simple. I wonder, though, if the price of connecting those modems to the phone line has kept the system from widespread use. I could throw together a modem with parts I have on hand and write a simple program for sending and receiving Baudot, but then I'm supposed to buy a $26 DAA to hook it to the telephone line! What's even worse, I have the equivalent of a DAA in my 1200-bps modem, and another in my Model 100, but they can't be used because nobody thought of adding an"auxiliary" jack (which I admit must partly be due to there being no prebuilt DAA, so the whole units are type approved under Part 68). A $2 (or whatever) solid-state DAA would probably help put some of those VIC-20s lying in closets to work for the deaf.

Michael Black
Montreal, Quebec

NEW **WICEZ8** In-Circuit Emulator

Z8

wytecwytecwytecwytecwytecwytecwytecwytecwytecwytecwytecwytecwytec

This NEW WICEZ8, containing the new Z86C12 CMOS ICE chip, is a complete In-Circuit Emulator for the entire Z8 microcontroller family, including the 86C91 and the 86E21 with 256 internal registers.

- **Comes** with window-oriented, user-friendly PC driver software for the **IBM PC, XT, AT or compatibles and permits real-time emulation up to 20 MHZ.**
- **Unique display windows monitor 30 programmable** memory loations or registers, 17 stack locations and are automatically updated when the user program is stopped or **single**-stepped.
- Addresses up to 128K bytes of memory and external data.
- Provides 32K emulation PAM mappable in 2K, 4K, 8K, 16K, or 32K blocks.
- Provides 8 hardware breakpoints which can always be displayed on the window.
- Symbolic debugger reads symbol files in the 2500 A.D., Microtek, Zax and Micro Computer Control formats directly.
- On-line assembler, disassembler, memory/register exam, compare, fill, move, search and modification.

038.4K baud rate for fast upload/download of files **in Tektronix Hex, Intel Hex and Motorola S record formats. (Download an 8K hex file in 5 seconds.)**

Regular Price: $1475 Introductory Price: **$995**

30 day Money Back Guarantee

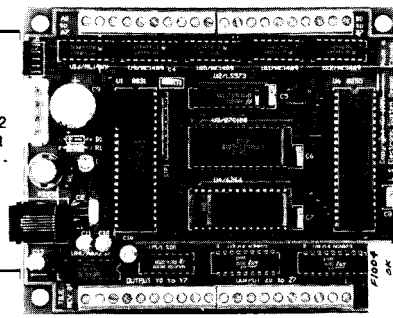
WYTEC

(708) 894-l 440
WYTEC COMPANY
Suite 140, 185C E. Lake Street, Bloomingdale, IL 60108

Reader Service # 169

DG31
8031 BASED CONTROLLER BOARD

- 8031 microcontroller running at 11.0592 MHz
- 16K EPROM and 8K SRAM JEDEC sockets
- 8255 parallel I/O, MAX232 for RS232C serial I/O port
- ULN2803A output drivers - 16 lines ±30 V input buffers - 20 lines. Screw terminals on 32 I/O lines
- Full schematic diagram, operating manual on disk



DG31 controller board assembled without EPROM $129.00

Controller KIT DG31K: PC board and all components except screw terminals, RAM and EPROM $70.00

EPROM 27C128 CMOS programmed with TILE firmware (Programmable controller with Real Time Clock) $20.00

EPROM 27C64 CMOS programmed with DG31M and disk (IBM format) with monitor program, modules source code listings and manual $30.00

HOME AUTOMATION and SIMPLIFIED SECURITY SYSTEM — complete project using TILE controller and low cost electronic modules (shipping included) $10.00

L.S. ELECTRONIC SYSTEMS DESIGN
2280 Camilla Rd. • Mississauga Ont. L5A 2J8 Canada

TERMS: **Shipping US/Canada $6. Check or money order please.**

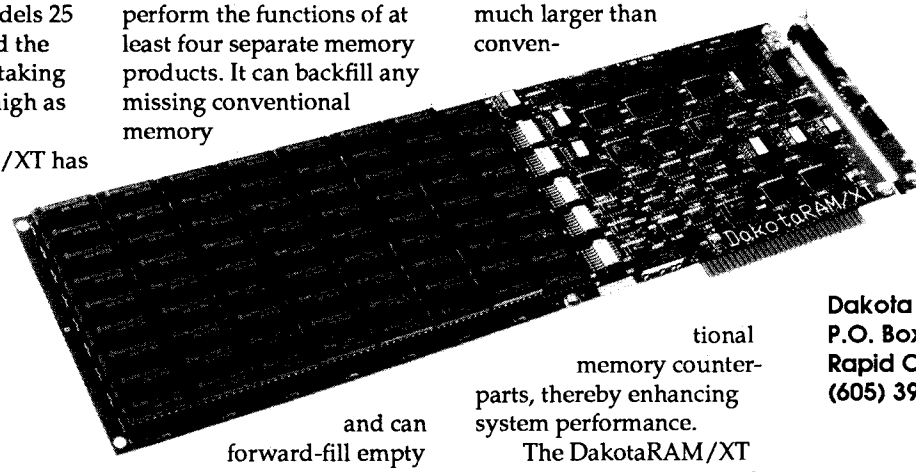
Reader Service #140

MEMORY ENHANCEMENT FOR OLD PC'S

Dakota Research Corporation has announced a novel way to extend the life of your old PC. The **DakotaRAM/XT** is a multitasking expanded memory board for IBM-compatible personal computers including the PC, XT, AT, and PS/2 models 25 and 30. It goes beyond the 640K barrier of DOS, taking memory capacity as high as 32 megabytes.

The DakotaRAM/XT has true multitasking capability with 16 hardware register sets; the ability to map memory anywhere in the 1-megabyte DOS space; direct memory access on board; and a fast, efficient expanded memory device driver. Programs can directly access these capabilities through standard EMS 4.0 function calls. Standard DOS programs can run concurrently under environments such as DESQview from Quarterdeck Office Systems.

The DakotaRAM/XT can perform the functions of at least four separate memory products. It can backfill any missing conventional memory and can forward-fill empty video memory space. The board can provide memory in the system area above video for running programs, device drivers, and TSR programs. Programs that recognize expanded memory have access to greater reserves for better performance. The RAM disk and print spooler can be much larger than conventional memory counterparts, thereby enhancing system performance.



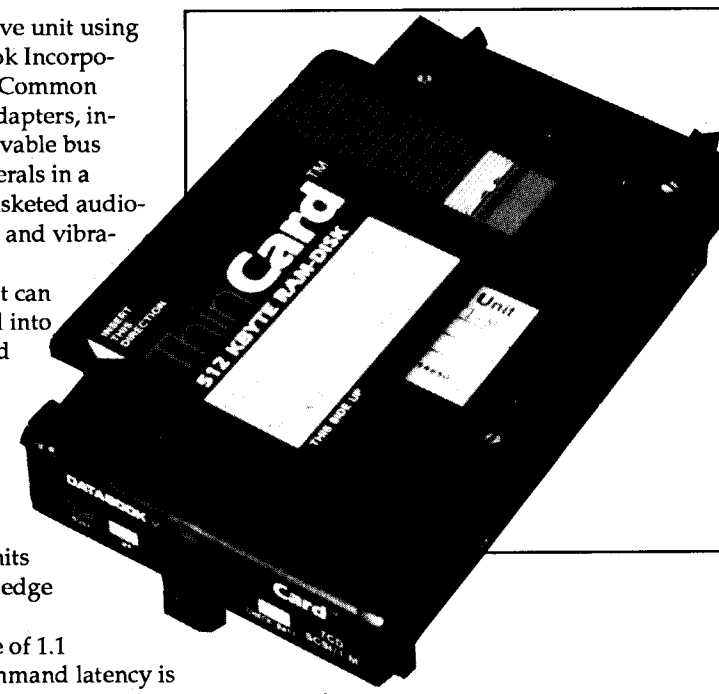
The DakotaRAM/XT comes with a minimum of 1 megabyte of memory and is expandable to 8 megabytes. Up to four boards can be installed in most computers for a total of 32 megabytes. Diagnostic software, expanded memory RAM disk, and expanded memory print spooler are included. The unit sells for **$695.00** with 1 megabyte of RAM. Additional RAM is available at $150.00 per megabyte.

Dakota Research Corp.
P.O. Box 40
Rapid City, SD 57709
(605) 394-8900

Reader Service #194

MINIATURE SCSI RAM-CARD DRIVE UNIT

A complete "Common Command Set" SCSI interface drive unit using removable IC memory cards has been introduced by Databook Incorporated. The **TCD SCSI/1 ThinCard** implements the full ANSI Common Command Set (CCS) for compatibility with most SCSI host adapters, including the Apple Macintosh. Selectable "SCSI ID" and removable bus termination resistors allow operation with other SCSI peripherals in a single SCSI host adapter. The unit is contained in a sealed, gasketed audio-cassette-sized box and is designed to withstand spray, shock, and vibration.



The storage medium is a credit-card-sized RAM disk that can contain from 512K to 8M bytes (when available). It is inserted into the drive in the same manner that a floppy diskette is inserted into a standard floppy drive. The card format is the same as that used by Databook's MS-DOS–compatible system as well as products from other manufacturers. Using the Read and Write Long commands, the Model TCD SCSI/1 can read and write data in any required proprietary format. This is useful when exchanging data with a dedicated system which stores the data in a format specific to its application. Drive units to support both the Epson 40-pin and Mitsubishi 50-pin card edge formats are available.

The TCD SCSI/1 features a maximum block transfer rate of 1.1 megabytes/second with CRC error detection. Worst case command latency is 300 microseconds with 250 microseconds being typical. Automatic error detection ensures data integrity, and SCSI parity is fully supported on data, command, status, and message transfers. Internal power monitoring ensures that card data will remain safe during power transients, and a built-in self test is run at every power-up.

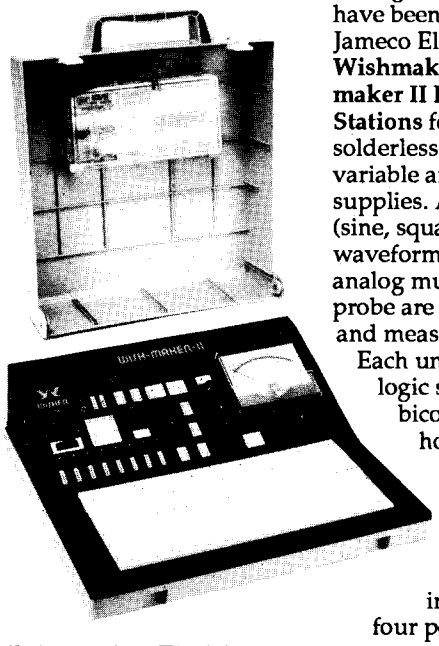
The drive unit, exclusive of the IC card and the SCSI cable terminators, uses only 0.8 watts maximum when active, and weighs under six ounces. Typical inactive power is 0.45 watts.

The TCD SCSI/1 lists for **$749**.

Databook, Inc.
Tower Bldg.–Terrace Hill
Ithaca, NY 14850
(607) 277-4817

Reader Service #195

PROTOTYPE DESIGN STATION



Comprehensive, portable breadboarding stations for analog and digital circuitry have been announced by Jameco Electronics. The **Wishmaker I** and **Wishmaker II Prototype Design Stations** feature a removable, solderless breadboard with variable and fixed DC power supplies. A multifrequency (sine, square, and triangular waveform) signal generator, analog multimeter, and logic probe are included for test and measurement functions. Each unit contains eight logic switches with bicolor LEDs and is housed in a sturdy, ruggedized case. The Wishmaker I, designed for analog prototyping, also features four potentiometers and a built-in speaker. The Wishmaker II, designed for digital prototyping, includes a pulse generator, binary-coded decimal (BCD) to 7-segment decoder/driver, frequency counter (1 Hz to 1 MHz), two push button momentary debounced logic switches, and a DB25 connector. Both units also include a wire jumper kit, probes, and manual.

The price of the Wishmaker I is **$199.95** and the Wishmaker II is **$249.95**.

Jameco Electronics
1355 Shoreway Road • Belmont, CA 94002
(415) 592-8097 • Fax: (415) 592-2503
Reader Service #196

ASSEMBLY LANGUAGE LIBRARY

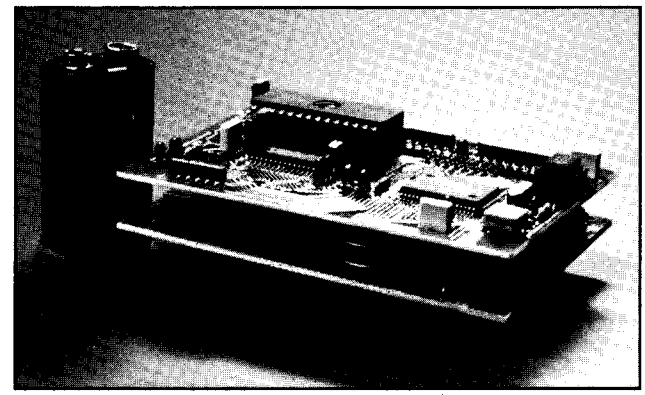
Program development can be accelerated and simplified with a set of assembly language routines introduced by Quantasm Corporation. The **Quantasm Power Lib** contains over 256 routines written entirely in assembly language. The Power Lib's design emphasizes speed and compactness, but the programmer maintains complete control and high-level functionality.

Some of the features include: overlapping windows with moving-bar menus and drop shadows (in 4K of code); fast, flicker-free screen routines; date/time calendar with date/time math, holidays, and so on; and extended-precision math. Over 75 string handling functions are included as well as sound effects, windowed keyboard input/editing, and file name parsing.

The Quantasm Power Lib comes with a comprehensive 200-page spiral-bound manual and over 3000 lines of example code. The software requires MS-DOS or PC-DOS version 2.1 or greater, 256K of RAM, and an IBM PC/XT/AT, PS/2, or compatible. The price is **$99.95**, or **$299.95** with the source code. The software is not copy protected and there are no run-time royalties.

Quantasm Corporation
19855 Stevens Creek Blvd. • Cupertino, CA 95014
(408) 244-6826
Reader Service #197

SINGLE-BOARD FORTH COMPUTER



A low-power, single-board computer, featuring a ROM-resident Forth language kernel and assembler, is available from The Saelig Company. The **TDS9090 Forth computer** uses the Hitachi HD63A03YFP CMOS microprocessor on a 4-inch by 3-inch board. Included on the board are two timers, two serial ports, and interrupts which are available via Forth instructions. Also included on the board are 30K of RAM for storing source code or data, 16K EPROM/NVRAM for firmware, 256 bytes of EEPROM, 35 I/O lines, a watchdog timer, and an expansion bus.

The 256 bytes of EEPROM keep vital data while the board is "sleeping." If the microprocessor crashes, the watchdog timer will reset the system and the application software will be reentered. This timer, which is external to the microprocessor, is reset by many Forth words.

The TDS9090 uses Fig-Forth with many extensions that are useful for a single-board system. For example, the number of microseconds taken by any Forth word can be accurately measured in real time. The language has been specifically implemented for the Hitachi microprocessor, and uses its facilities wherever possible.

Two modes of operation are available. An interactive mode is useful for debugging, and typing a Forth command causes the action to occur immediately. The second mode is noninteractive and features a full-screen editor. An on-board compiler allows code to run up to 10 times faster than equivalent BASIC code. Application code can be put in EPROM to create a stand-alone system, or used as a turn-key system from the nonvolatile RAM supplied.

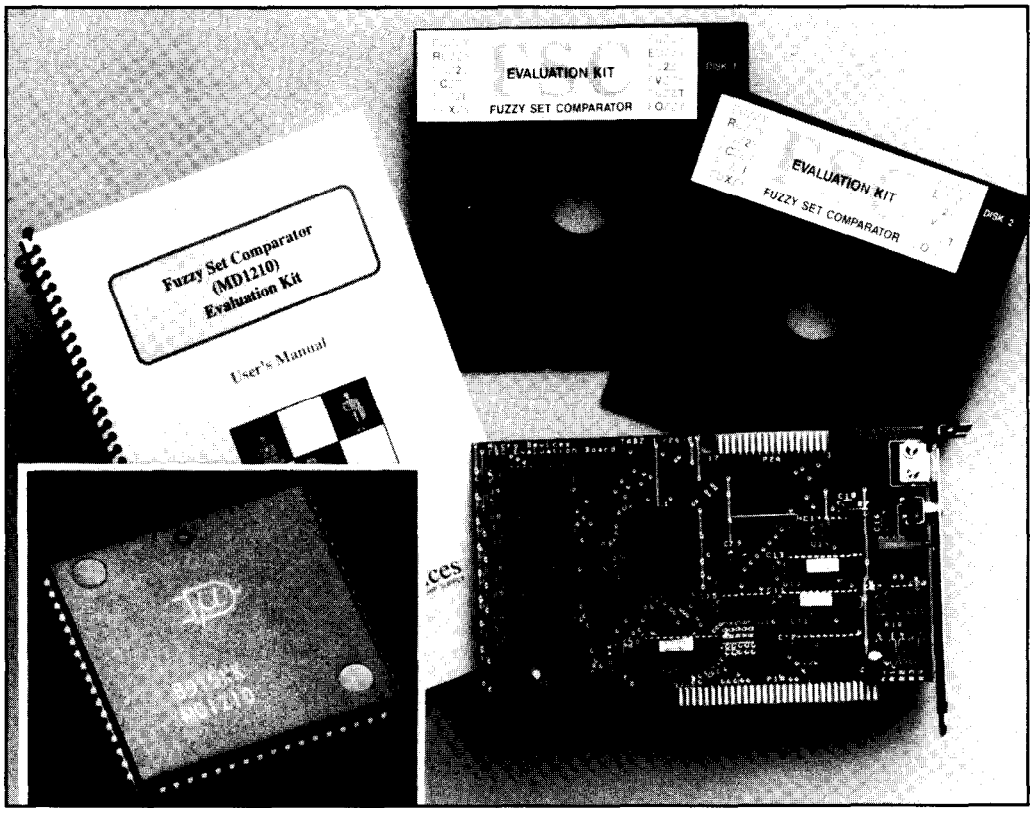
For data logging, the **TDS9095** plugs directly into the TDS9090. It features a 10-channel, 8-bit ADC; 6-channel DAC; 128K bytes of RAM; and a nonvolatile date/time clock. Software provided simplifies the addition of an LCD display or matrix keyboard. Data can be sent or received via the two RS-232 ports.

The price of the TDS9090 is **$399.00** and includes manual, NVRAM, PC software, and sockets. The TDS9095 costs **$479.00** and includes manual, PC software, and sockets.

The Saelig Company
1193 Moseley Road
Victor, NY 14564
(716) 425-3753
Fax: (716) 425-3835

Reader Service #198

FUZZY SET EVALUATION KIT



Integrating artificial intelligence into pattern recognition applications has been simplified with the introduction of a Fuzzy Set Comparator (FSC) from Micro Devices. The MD1210 FSC is a CMOS VLSI chip that does pattern recognition by very fast comparisons of serial bit streams. It incorporates a digital hardware neural network to do the comparisons, and can be used in any digital computer.

The MD1210 can do pattern recognition much faster than software-based schemes by using its neural network to process "fuzzy data" (inaccurate, noisy, or otherwise variable data). It is capable of comparing eight unknowns to one known, or one unknown to eight knowns, and delivering a decision in as little as 250 nanoseconds. The device's expandable architecture allows simultaneous examination of up to 256 fuzzy sets without sacrificing speed.

Comparison functions are done using either Linear or Hamming distance measurement. Since field lengths greater than one bit are being processed in most cases, Linear is usually used. Under best-case conditions, the MD1210 can learn or compare data at a rate of 20 MHz. Applications include target acquisition, CAM systems, image or voice recognition, and robotic control.

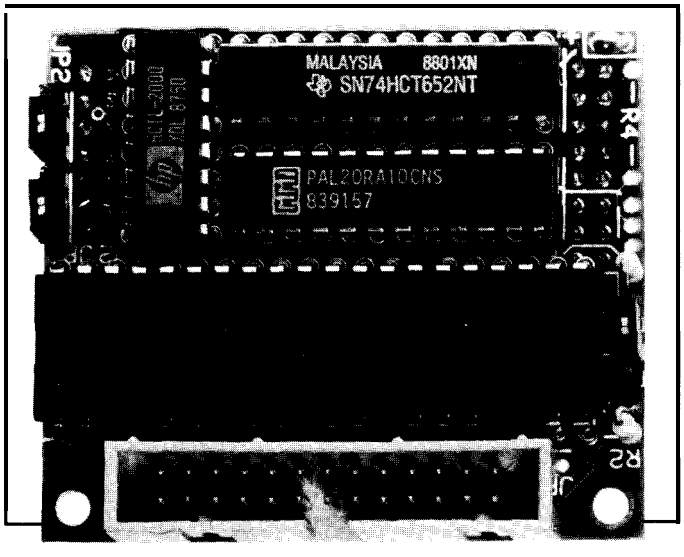
The MD1210 Evaluation Kit is designed as an inexpensive tool for providing real-time operation demonstrating the various features and operating modes of the MD1210 FSC. In addition to the MD1210, the evaluation board contains a video frame grabber, eight pattern memories, and the necessary circuitry for installation in an IBM PC/XT/AT or compatible. Also included in the kit is evaluation software providing a menu-driven program for accessing all of the MD1210's available functions and complete documentation.

The price of the MD1210 FSC Evaluation Kit is $250.00.

Micro Devices
56958 Beggs **Road**
Orlando, FL 32810
(407) 299-0211
Fax: (407) 290-0164

Reader Service #199

STEPPER MOTOR CONTROLLER



A programmable motion controller with up to eight axes of control has been announced by Precision Micro Control Corp. The DCX-MC160 is a stepper motor controller with encoder interface for position verification. The unit features 32-bit position resolution and a step range from 20,000 to 0.02 steps/sec. Outputs include direction and pulse, or pulse left and pulse right. Position and velocity modes are provided along with an internal trapezoidal velocity profile generator.

Velocity, acceleration, and initial step rate are programmable, and the unit features open-loop operation and an encoder interface with index pulse input. Additional input/output lines include home, limit left, limit right, jog, and stopped.

The DCX-MC160 plugs into a DCX-PC100 (PC bus) or DCX-VM100 (VME bus) motherboard to produce an off-the-shelf programmable motion controller. Pricing was not available.

Precision Micro Control Corporation
3555 Aero Court
San Diego, CA 92123
(619) 576-8058
Fax: (619) 565-1186

Reader Service X200

VISIBLE INK

answers: clear and simple

Letters to the INK Research Staff

KEEP THOSE LEGOS MOVING

With my son hooked on Legos and myself on robotics, it was natural to combine the two with Lego Technics. The Lego Educational Department makes an Apple //e interface called the Technic Control II consisting of a slot card pack for $145 and an interface box and transformer for $170. The card plugs into one of the Apple's seven expansion slots and runs off Turtle-language software.

I've found a cheaper and easier way to do the same thing using the 16-pin internal I/O game port with its four annunciator output pins and paddle and switch inputs. I use the standard reversing motor circuit illustrated in "DC Motor Controls" [Ciarcia's Circuit Cellar, Volume III] using two annunciator outputs for each bidirectional motor.

My problem is this: Lego Technic motors are designed for 4.5-6 VDC, provided by a four-C-cell in-line battery pack. Using 2N2222 single transistors in the reversing circuit, only a fraction of the power is delivered to the motor, and the transistor overheats and bums out. Using piggyback 2N2222s switches usable power to the motors, but they still run hot. **Using TIP31** power transistors (Ic=3A) and heat sinking solves the heating problem, but they seem to need more current at the base than the annunciator can provide, forcing me into a Darlington arrangement of a 2N2222 feeding a larger power transistor.

How do I **measure** usable power outputs of the circuits I build, to compare quantitatively one against the other (right now I estimate motor RPM)? How would I estimate stalling amperage demand for a Technic motor? Is the Darlington scheme above the best way to switch the highest percentage of power from the battery to the motor?

Carl **Bakay**
Harvey, LA

People are divided into two classes: Lego fanatics and the rest. Fortunately, we are in the same class! You've done a good job with the Lego Technic motor circuit and a few more tips should have your setup humming...it's always moreinstructive to do things yourself, even if **you** *smoke a few transistors along the way.*

In order to explain how to makea motor *driver work, we need to go into a little transistor theory. Although we think of the transistors as digital switches, they are analog devices and we must take that info consideration when designing circuitry!*

Transistors have a myriad of characteristics, but the four most important for this application are the maximum collector current rating (V_{CEO}), the DC current gain (h_{FE}), the collector-emitter saturation voltage ($V_{CE(sat)}$), and *the maximum power dissipation. Table 1 summarizes* these *ratings for a few parts.*

TYPE	MAX CURRENT	GAIN	SATURATION	MAX POWER
NPN				
2N2222	800 mA	30	1.6 V	0.8 W/ 3W
TIP31	3 A	10	1.2 V	2 W/ 40 W
TIP120	3 A	1000	2 v	2 W/ 65 W
PNP				
TIP32	-3 A	10	-1.2 V	2 W/ 40 W
TIP125	- 3 A	1000	- 2 v	2 W/ 65 W

NOTE: POWER RATINGS AT 25°C AMBIENT / 25°C CASE TEMPERATURE

Table 1 —*Four important transistor characteristics include maximum collector current, DC current gain, collector-emitter saturation voltage, and power dissipation.*

Those 2N2222 transistors have a maximum collector current raring **of** *800 mA. While that's a lofforan electroniccircuit, it's next to nothing for even a small motor. A motor from my junk box drew 500 mA at 5 V with no load and about 3 A when I grabbed the shaft; thepowersupplycurrent-limited at that level, so the actual value is much more. If your motors are similar, you* can see *why* 2N2222 transistors fail quickly: even a *small load on the motor will push them well beyond their maximum rating.*

It is quite simple to measure the motor currents. You *need a multimeter that can measure a few amperes of DC current, which may cost $20 at the local Radio Shack. Connect the batteries, motor, and meter in series, then read the* meter **to** *find the no-load current. If you hold the shaft stationary, the meter will read fhe stall current.*

The TIP32 is a reasonable choice for a small motor driver, becauseitcanhandleup to 3 A of current. The next step is to make sure *that the transistor will act as a real switch. I'll start by discussingasingle TIP31 connected as in Figure 1, which is* half of *the circuitry you used in your driver.*

The DC current gain rating is simply the ratio of output current (through the collector) to input current (through the base). The TIP31 has a gain of 10, which means that the base current must be300 mA (0.3 A) to cause3 A of *collector current.*

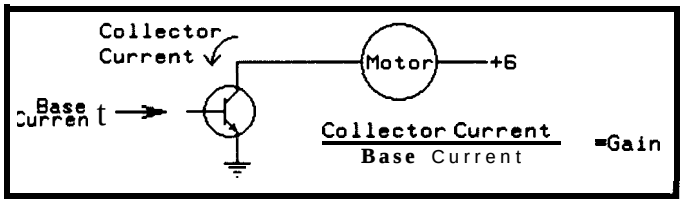


Figure 1 -The TIP3 1 has limited application controlling motors.

The current gain is actually a function of the collector current and the fabrication process, so it is usually specified as a range; the value for TIP31 transistors may actually be 10 to ZOO!

The key to using *a transistor as a switch is to "turn on" the collector by forcing current into the base terminal. When a transistor is "turned on" (or "saturated"), the collector voltage stays at $V_{CE(sat)}$ regardless of theactual current. For TIP31 transistors driven with 300 mA of base current, the collector saturationvoltagewill be under 1.2 volts as long as thecollectorcurrent is less than 3 A.*

Because the collector can handle any current up to the limit set by the basecurrent times the current gain (or the transistor's maximum rating!), theactual current will beset by theexternal circuit. If you measured a no-load motor current of 500 mA, the collector current will be500 mA because the motor won't permit any more current to flow through the circuit. As you load the motor, of course, it will conduct more current.

The power dissipated by the transistor is set by the product of the collector current and the collector voltage. In this case, it will be500 mA times 1.2 volts, or about 600 m W. *In actual fact, the saturation voltage will be lower than the rating because the current is so low, so the actual power dissipation will be lower.*

Digital logic circuits can't *supply 300 mA to drive the TIP31, though. A standard LSTTL bus driver, such as the 74LS244, can supply perhaps 40 mA to a transistor base, which is too little to turn the TIP31 on completely.*

What happens in this case is that the collector voltage rises (it is no longer "saturated") until thecurrent through thecircuit falls to thelevelset by the basecurrent. Theproductof thecurrent and voltage still gives the transistor power dissipation, which may be 400 mA times 4 volts: 1.6 watts!

Under normal conditions a load applied to the motor would increase thecurrent by reducing themotor's internal resistance. With the transistor out of saturation, however, reducing the motor's resistance increases the voltage applied to the transistor's collector. In the limit, the TIP31 will dissipate 400 mA times 6 volts: 2.4 watts.

Zf the TIP31 isn't on a heat sink, it can dissipate only 2 Wat anambient temperature of 25°C. As youmightexpect,it will get pretty hot! You can see why the 2N2222 transistors burned out at the first chance...

The solution is to put a predriver transistor between the digital output to ensure that the TIP31 gets enough basecurrent. You can do this with a discrete 2N2222 transistor "piggy-backed" on a TIP31, but a TIP120 puts two matched transistors in a single package for about a buck. As you can see from *Table 1, the current gain is over 1000, which means that you* need *only 3 mA of base current to get 3 A of current at the collector.*

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 lax

FEATURES

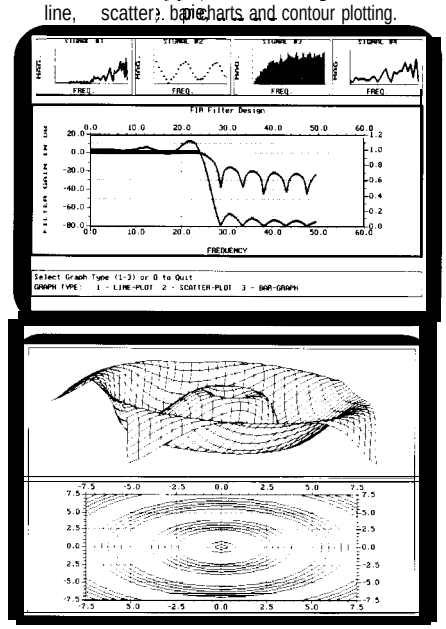
100% Royalty Free

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, bar, pie charts and 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, Power Spectrum, FIR Digital Filtering

Matrix Math Real and complex

Complex Number Arithmetic

Eigen values and vectors - Cyclic Jacobi

Integration Simpson's method

Differential Equation Runge-Kutta-Fehlberg

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

PO Box 26, Newton, MA 02164 USA

Tel. (617)965-5660 FAX (617)965-7117

Reader Service #154

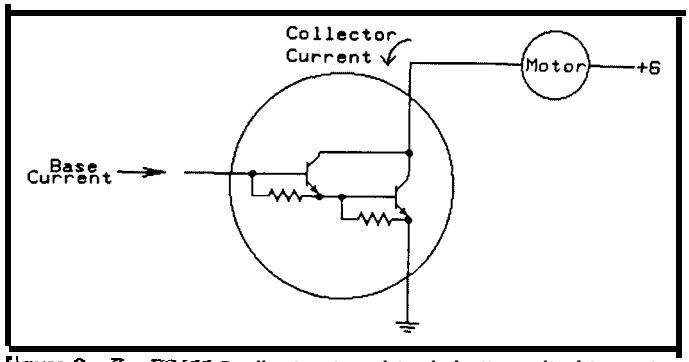


Figure 2—*The TIP120* **Darlington transistor is better suited to motor control.**

Figure 2 is similar to Figure 1, because the Darlington pair is packaged in the same three-lead package (the resistors shown are inside, out of view!) From our viewpoint, it's just a transistor wifha remarkablyhighcurrenf gain. Thereareofherfacfors that come info play for other designs, but we don't have to worry about them here.

The tradeoff is that the collector saturation voltage is now 2 volts instead of 1.2 volts, so a little more power goes info the transistor: 1 waft at 500 mA instead of 600 mW. The power rating is 2 Win *free air, so you can see that a heat sink is a good idea.. .particularly if you intend to put any loads on the motor!*

Also, fhevolfage between the base and emitter terminals increases foabouf 2.5 volts becauseif includes two diodedrops (the two base-emitterjunctions). Ordinarily you allow about onevolf for this drop when figuring the base circuit resistance, so you may need to make a few changes. As with all parameters, this voltage increases with current, so if will be lower if you don't drive the transistor very hard.

Figure 3 shows a revised motor driver circuit that incorporates some *improvements over the original design. Note that the "upper" devices in each pair are TIP125 transistors. These are PNP Darlingtons with specs similar to NPN TIP120s; indeed, TIP120 and TIP125 devices are "complemenfay" transistors because they are well matched.*

Although fhelogicgafes in your Apple are digital and we are using the transistors as digital switches, you must remember that "digital" circuits are really "analog"-at heart. One point that often gets forgotten is that you can't connect excessive voltages to logic gates without suffering dire consequences. Figure4 includes four 7407 open-collector buffers that translate from the TTL logic levels to the 6-volt DC motor switches.

There are two logic inputs to the driver circuit: one selects the motor direction, while the other turns the motor on and off. The hardware translates these digital bits info the appropriate transistor base currents and ensures that only one transistor on each side of the motor is turned on *at a time.*

The original design gave you individual control for each of the four driver transistors, which meant that an errant program could turn both transistors driving one side of the motor on at once, thus shorting the power supply to ground!

A few examples may clarify the circuit's operation. First, when the RUN input is low, the two NAND gate outputs are high, so the outputs ofallfour 7407 gates are also high. This will

We got you covered...

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

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

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

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

Paradigm LOCATE $295.00
Turbo Debugger Interface $195.00

PARADIGM SYSTEMS INCORPORATED

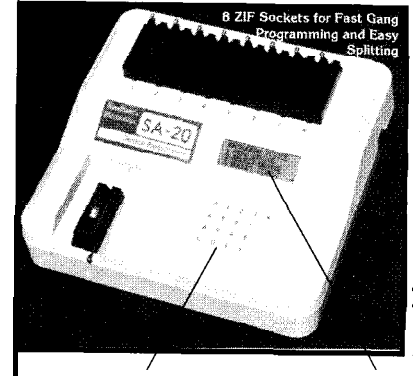
Satisfaction Guaranteed
Orders: (800) 537-5043
Technical Support: (508) 478-0499
BIX: join paradigm
Visa/Mastercard/C.O.D. Accepted

Paradigm Systems
P.O. Box 152 Milford, Massachusetts 01757
Turbo Debugger is a registered trademark of Borland International

Reader Service #150

EPROM PROGRAMMERS

Stand-Alone Gang Programmer $750.00



- Completely stand-alone or PC driven
- Programs E(E)PROMs
- *1 Megabit of DRAM*
- *User upgradable to 32 Megabit*
- *.3/.6" ZIF socket RS-232, Parallel In and Out*
- 32K internal Flash EEPROM forea: firmware upgrades
- *Quick Pulse Algorithm (27256 in 5 sec, 1 Megabit in 17 sec.)*
- 2 year warranty
- *Made in U.S.A*
- Technical support by phone
- Complete manual and schematic
- *Single Socket Programmer also available. $550.00*
- Split and Shuffle 16 & 32 bit
- 100 User Definable Macros, 10 "se Definable Configurations
- Intelligent Identifier
- Binary, Intel Hex, and Motorola S

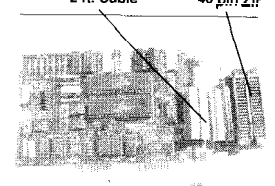
20 Key Tactile Keypad (not membrane) 20 x 4 Line LCD Display

Internal Programmer for PC $139.95

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

- Reads, verifies, and programs **2716.32,** 32A, **64.** 64A, 128, 128A, **256** 512 **513** 010 011 301 27C2001, MCM 68764, 2532
- *Automatically sets programming voltage*
- Load and savebuffer todisk
- Binary, Intel Hex, and Motorola S formats
- *Upgradable to 32 Meg EPROMs*
- *No personality modules required*
- 1 year warranty • 10 day money back guarantee
- Adapters available for **8748. 49, 51,** 751, 52, **55,** TMS 7742, 27210, 57C1024, and memory cards
- Made in USA

2 ft. Cable 40 pin ZIF



NEEDHAM'S ELECTRONICS
4539 Orange Grove Ave• Sacramento, CA 95841
Man Fri Barn-5pm PST

Call for more information
(916) 924-8037
FAX (916) 972.9960

C.O.D. MasterCard VISA

Reader Service #147

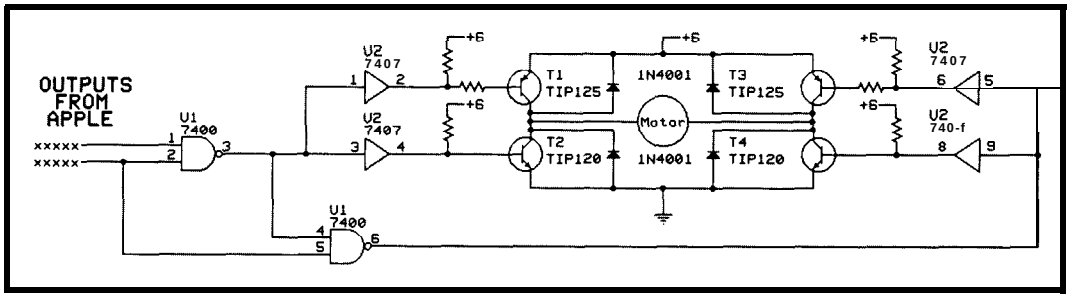


Figure 3—*The TIP* 120 *and TIP* 125 work together to *provide an easy-to-use motor interface.*

turn both TIP120 transistors on, because their base terminals will conduct about 3 mA through the resistors. Both TIP125 transistors are off, because their base terminals see the same 6 volts as their emitters and thus cannot draw any current.

If you raise the RUN input with the DIR input low, Ula stays *high and U1b* goes *low,* so *T2 remains on and T4 goes off. The output of U2c is low, so the base of T3 conducts about 3 mA. That turns T3 on, which applies voltage to the right-hand motor terminal. Because T2 is on, the left-hand terminal is grounded and the motor begins to spin.*

Now, if you raise the DIR input, T1 and T4 go on while T2 and T3 shut off, and the motor reverses because the applied voltage changes direction. As you probably know, *a* DC *motor makes a perfectly good generator* **if you spin** *the shaft, so the motor will "buck" the applied voltage in this case. Figure 3 includes four diodes to handle this current; the voltage at any motor terminalcannot exceed 6voltsorgo belowground, norare the transistors exposed to reversevolfagesfromgeneraforacfion.*

Finally, what battery voltage is getting to the motor? Because the motor is in series with two transistors, you subtract thecollector saturation voltages from the battery *voltage fofind what* **goes** *to themotor. Thespecs would have* **you** *believe that the transistors eat up 2 volts each, so* only 2 volts *are left!*

In *practice, the specs* are *conservative, so* the *motor* will *actually see about four volts. You can use your new multimeter to measure the motor voltage under various loads and see how accurate the specs are.*

You could also use relays rated for the maximum motor current instead of transistors. You may need a transistor driver circuit to power the relay coils.

You should *now* have enough information to build a *Lego* motor *driver circuit to end all circuits. Tell us how it works out.*

IRS

201 Very Useful
202 Moderately Useful
203 Not Useful

ONE HOUR PROTO-TYPING !

A ***new*** circuit board proto-typing method that takes full advantage of existing CAD/Plotter circuit development systems.

Advantages

- ❑ Fast **one hour prototype creation;**
- ❑ **No hole drilling;**
- ❑ **No photo developing errors;**
- ❑ **No dangerous ultra violet light;**
- ❑ **Double or single sided boards;**
- ❑ **Low cost, less than 50¢ per. sq. in.**

EACH KIT COSTS -- $70.00
Enough material to make 1 sq. foot of double sided boards.

To order call T.O.A.D. Inc.
1-800-323-8623.

Reader Service #164

MACINTOSH CROSS ASSEMBLERS

µASM™ ***NEW!! Version 3.0!***

*TEXT EDITOR, CROSS ASSEMBLER, AND COMMUNICATIONS FACILITY IN A COMPLETE INTEGRATED DEVELOPMENT ENVIRONMENT
• MACROS
. CONDITIONAL ASSY
. LOCAL/AUTO LABELS
. SYMBOL TABLE CROSS REF
. S OR HEX FILE OUTPUT DOWNLOADS TO MOST EPROM PROGRAMMERS

us **$149.95 EACH PLUS S/H***

AVAILABLE FOR MOST 8-BIT MICROPROCESSORS. CALL OR WRITE FOR TECHNICAL BULLETIN. 30 DAY MONEY BACK GUARANTEE. MCIVIAE.

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

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

Reader Service #142

FEATURE ARTICLE

Implementing A ComeFrom Statement

J. Conrad Hubert

Discover Where Your Code Has Been

There's a long-standing lament among BASIC programmers: 'If I only had a ComeFrom statement, I could get this code to work!" As it turned out, I needed a ComeFrom statement to implement a minimal-parts-counton/off switch in a battery-powered embedded system. Since the membrane keypad I selected for microprocessor input was not directly useful as a main power switch, I chose not to have a conventional on/off switch at all. The decision was possible because CMOS microprocessors like the 80C51 and DS5000 have a software-invoked "idle" mode which dramatically reduces their power consumption. The on/off switch simply toggles the microprocessor between normal operation and idle mode. *[Author's Note: If you use the 80C51 and are not familiar with the Dallas Semiconductor 055000, do yourself a favor and get their development system. The processor comes in 8-,12-, and 16-MHz versions with 8K or 32K bytes of battery-backed SRAM, and has a serial loader in ROM. A time-of-day clock is optional.1*

```
        ORG     0003H   ; EXO vector location.
        AJMP    TRICK   ; Routine won't fit in 8 bytes,
                        ;   so jump to the TRICK subroutine.
```

Listing **1 -The Vector *Table* Jump.**

The code in this article illustrates, in three parts, how the mode switch is performed. The three segments are: a vector table jump, a trick to alter the return address for the interrupt, and a subroutine to toggle the sleep state.

When pin 12 of an 80C51 is pulled low, it generates external interrupt zero (EX0). The processor responds to this interrupt by suspending whatever it is doing, pushing the address of the last instruction it executed onto the stack, and jumping to EX0's vector address. The vector address for EXO is 003H, and the next interrupt vector (EX1) is OOBH. If the interrupt handler is eight bytes or fewer, the routine can reside right in the vector table, otherwise a jump to another location must occur.

Since the user may decide to turn the device off at any time, interrupt EXO could occur anywhere in the execution of the code. Part of the definition of an interrupt-to resume execution at the instruction immediately following where it was suspended—was not compatible with my inten-

```
TRICK MOV   SP,#7         ; If EXO occurred while there was
                              even one pending return from a
                              previous call, all unreturned
                          ;   addresses could, eventually, cause a
                          ;   stack overflow. I picked 7 because
                              it is the contents of the stack
                          ,   pointer after a power-up reset.
      MOV   DPTR,#TOGGLE  ; DPTR is the only 16-bit register.
                          ;   It receives the address of the
                          ;   subroutine toggle.
      PUSH  DPL           ; Put that address on top of stack,
                          ;   low byte first,
      PUSH  DPH           ; followed by high byte.
      RET1

; When the subroutine TRICK is done, the RET1 instruction
; causes execution to resume at the instruction immediately
; following the last instruction executed prior to servicing
; the interrupt. Since the address of TOGGLE is now on top of
; the stack, execution resumes at the address TOGGLE+1.
```

Listing ***2-A** trick to alter the* **return** *address* **for *EXO.***

tions. What I needed was a way to toggle the sleep state when the interrupt occurred and then, **depending on** the new state, either execute the initialization code or put the microprocessor to sleep.

Of course, the interrupt handler could have simply jumped to the appropriate subroutine and continued as though the interrupt had never occurred. Unfortunately, this plan will eventually result in a stack overflow because the return from interrupt (RETI) instruction, which pops a return address from the stack, is never executed. I needed to trick the processor into executing a specific subroutine after "coming from" an interrupt. The code I used, shown in Listings l-3, assumes EXO is enabled and is set for edge-triggered operation.

All of this is applicable to external interrupt 1 (EX1) as well as EXO. Just use EX1's vector table to jump to the location of TRICK and use pin 13 for the interrupt signal. In fact, I used EX1 for another toggle because it obviated polling time in a section of speed-critical code.*

```
TOGGLE  NOP                ; Not executed. A place holder nec-
                           ;   essary because execution begins at
                           ;   TOGGLE+1.
        CPL     PSW.5      ; Complement Processor Status Word
                           ;   bit 5, PSW.5 is a user flag which
                           ;   stores the sleep state.
        JB      PSW.5,INIT ; If asleep, jump to initialization
        MOV     PCON,#1    ; Otherwise, setting bit 0 of the
                           ; Power Control Register invokes
                           ;   idle mode.
```

listing 3-A subroutine to toggle the On/Off state via EX0.

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

IRS

205 Very Useful
206 Moderately Useful
207 Not Useful

Call for Manuscripts

CIRCUIT CELLAR INK is looking for quality manuscripts on software for embedded control, software applications, advanced algorithms, 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**

GET TO WORK!

A New Project

Our line of macro Cross-assemblers are easy to use and full featured, including conditional assembly and unlimited include files.

Get It To Market--FAST

Don't wait until the hardware is finished to debug your software. Our Simulators can test your program logic before the hardware is built.

No Source!

A minor glitch has shown up in the firmware, and you can't find the original source program. Our line of disassemblers can help you re-create the original assembly language source.

Set To Go

Buy our developer package and the next time your boss says "get to work", you'll be ready for anything.

Quality Solutions

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

BROAD RANGE OF SUPPORT

- Currently we support the following microprocessor families (with more in development):

Intel 8048	RCA 1802,05	Intel 8051	Intel 8096
Motorola 6800	Motorola 6801	Motorola 68HC11	Motorola 6805
Hitachi 6301	Motorola 6809	MOS Tech 6502	WDC 65C02
Rockwell 65C02	Intel 8080,85	Zilog 280	NSC 800
Hitachi HD64180	Motorola 68000,8	Motorola 68010	

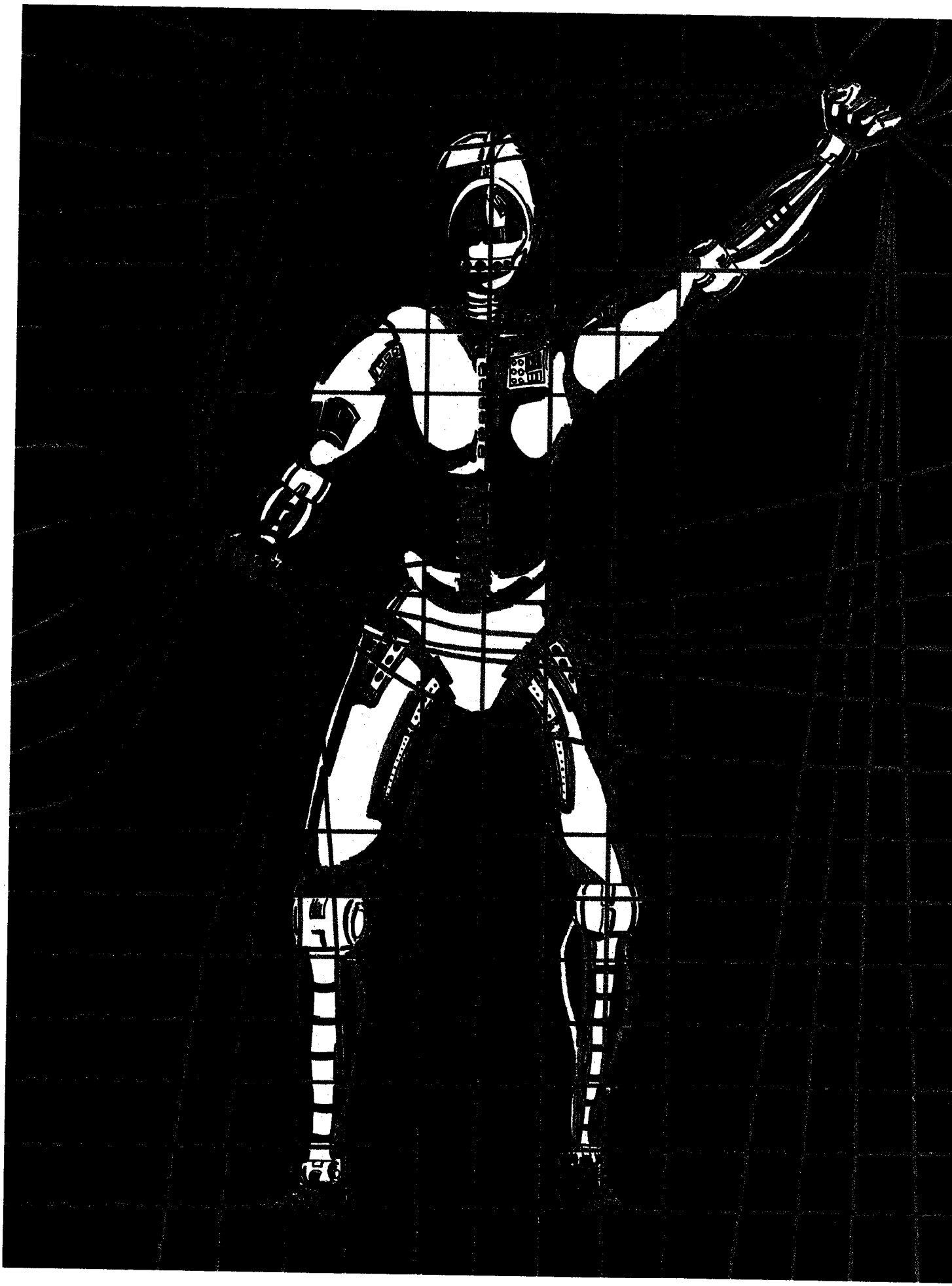
- All products require an IBM PC or compatible.

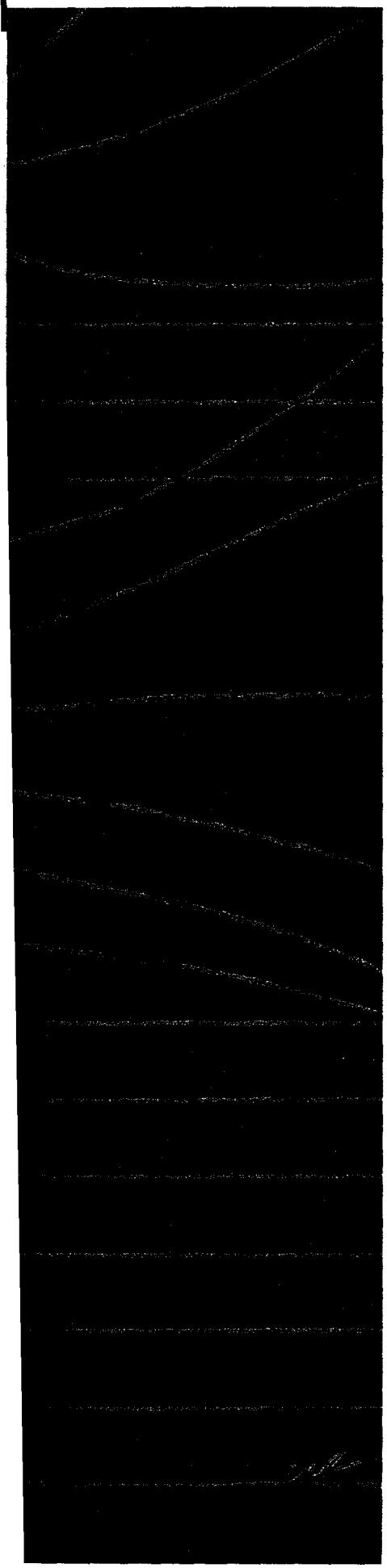
Cross-Assemblers as low as $50.00
Simulators as low as $100.00
Cross-Disassemblers as low as $100.00
Developer Packages as low as $200.00
(a $50.00 Savings)

So What Are You Waiting For?
Call us

PseudoCorp
Professional Development Products Group
716 Thimble Shoals Blvd, Suite E
Newport News, VA 23606
(804) 873-1947

ader Service #153





Robotics and Artificial Intelligence

Modeling Synthetic Actors and Real- World Interactions

FEATURE ARTICLE

Chris Ciarcia

When I first sat down to write this paper I, like everyone else, fought the typical first-line battle of "how do I start the damn thing?"

In the background of my mind I juggled ideas and concepts of how I would compose this paper. And then it occurred to me that my current effort in many ways reflected how each of us are victims of our own imaginations. I couldn't shake the idea that the implied promises of fantastical scientific capabilities envisioned in such epics as Star Wars and Star Trek have, in many ways, finally caught up with us. The view of the future displayed in these cinematic wonders is one of great intelligence. Not the expanded intelligence of man, but rather the extension of intelligent behavior to every conceivable object within man's existence. The level of sophistication promised by these "futuristic devices" is still far beyond our current capabilities.

This article will give you some insight into the complexity of artificial intelligent behavior. To that end, I'll discuss some of the basic concepts behind robotics and artificial intelligence with special emphasis on their application to the interaction of robotic entities with objects in a defined environment. Simple studies of this sort can be undertaken on your PC by modeling various environments and actors who interact with those environments, with the ultimate goal of becoming aware of the types of obstacles which must be overcome for "realistic behavior."

We will employ the tools of animation and a synthetic actor: a computer construct which is a simulation of some entity, be it a robotic arm, a human, or some functional machinery. I'll briefly discuss synthetic actor design and animation in order to give you a flavor of how animated sequences are usually generated. Next, I'll discuss motion **planning and** intelligent activities, obstacle avoidance, and object manipulation since these are more specific to robotics and our free-style animation. ***[Editor's Note:*** Software for *this article is available from the Circuit Cellar BBS and on Software On Disk #15. For downloading and ordering information, see page* 77.1

SYNTHETIC ACTORS

Within our context, the synthetic actor is defined as a simulated character, be it a robot or a human. It emulates the functional appearance, behavior, and environmental response which its real-world counterpart would display under similar circumstances. It is directed by task-level commands that enable it to be conscious of its environment, move about, communicate, manipulate objects, and modify its own personal appearance when conditions demand it.

The realization of a truly effective synthetic actor is extremely difficult. It has become an interdisciplinary endeavor (see Figure 1) which integrates aspects and methods from animation, mechanics, robotics, physiology, psychology, and artificial intelligence. As such, it has generated exten-

sive research within the following areas:

Image Synthesis Modeling-the physical aspects of the actors: shapes, colors, textures, reflectances, and so on.

Complexityand Realism of Motion—descriptions of limb and body motion as well as their deformations during motion; robotics for task planning, object grasping, and obstacle avoidance.

Methods in Artificial Intelligence—to create consciously interactive actors that can experience and learn from an environment and make sensible decisions; communications in the form of information exchange or the development of artificial dialogue control; and the study of human or mechanical systems behavior in order to capture true representations of the real-world original.

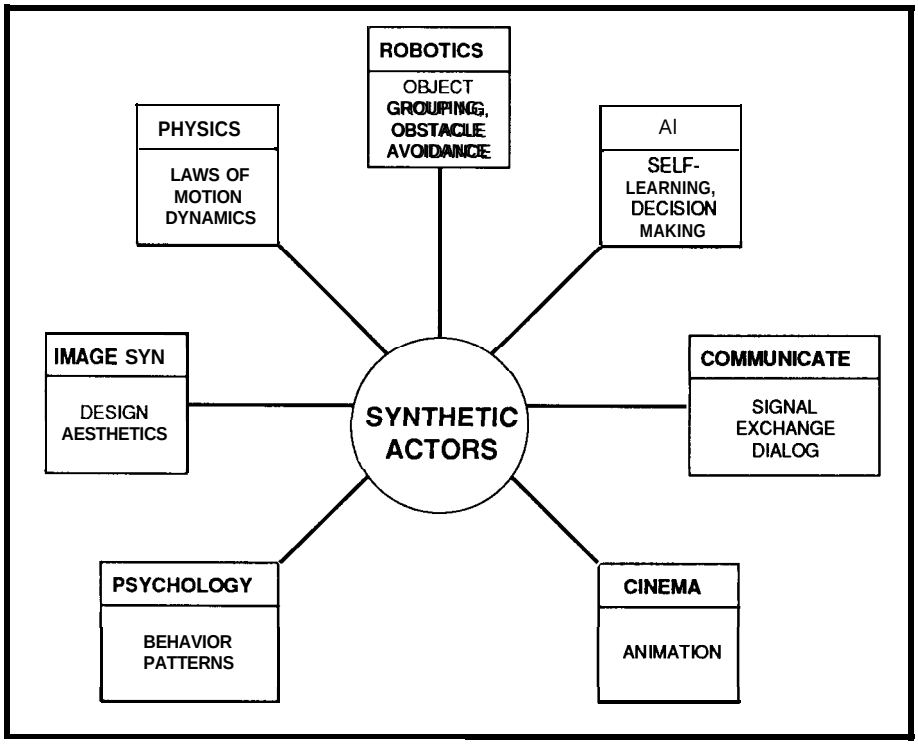


Figure 1 -*Synthetic actors; an interdisciplinary undertaking.*

MODELING

To create natural motions and environmental interactions, you must take into account the geometry, physics, and behavior of both the synthetic actor and the components of the associated environment. Only a system based on these six factors can really work. To date, four widely accepted techniques are in general use: shape interpolation, parametric interpolation, kinematic algorithmic animation, and dynamic algorithmic animation. Shape interpolation is based on a sequence of key frames and consists of the automatic generation of intermediate frames, called in betweens. The quality of the resultant animation depends highly on the number of key frames, the type of interpolation law, and the number of points evaluated.

Parametric interpolation is based on the interpolation of parameters of the model of the synthetic actor itself. Each key frame is specified by an appropriate set of parameter values. Parameters are then interpolated, and images are finally individually constructed from the interpolated parameters. The quality of the resultant animation sequence depends on the number of key values and the number of parameters.

Algorithmic animation, or procedural animation, is achieved by the algorithmic description of the physical laws which determine how motion and interaction takes place. With such an approach, any kind of law may be applied to the parameters. For example, the variation in a joint angle can be controlled by kinematic laws as well as dynamic laws. Kinematic procedures define motion in terms of the displacement, velocity, and acceleration of individual points, while dynamic procedures involve the use of a set of forces and torques to determine motion. In general, the dynamic application is more realistic than the kinematic model, but pays a price in its size and complexity. A complete dynamic model is usually too large and expensive to be practical.

FIGURING THE FACTORS

The choice and classification of the animation sequence best suited to a specific application depends on several factors. It depends on whether the assigned coordinate system is two orthreedimensional;whethertheform of the synthetic actor's body is a stick, surface, or volume model; whether the choice of motion model is kinematic or dynamic; and how the movements are specified-whether it uses guiding-key-frame interpolation, program-level algorithmic animation, or a task-level command procedure of predefined or computable motions. Synthetic actor animation is therefore averycomplicated task;mosthumans or robotic systems being modeled are of irregular shape and difficult to define.Specificationandcomputation of the synthetic actor's movements nontrivial, especially since complex articulations of limbs and surface features have been known to involve more than 200 degrees of freedom.

As a result of this complexity, the total animation problem has been divided into three basic components:

- *modeling* the synthetic actor's body.
- *specification and computation* of movement.
- rendering-total image synthesis of the sequence, shading, and hidden surface removal.

THE SYNTHETIC ACTOR'S BODY

The first step in our animation sequence is generating the body of

our synthetic actor. This is accomplished by specifying a geometric model which describes each of the body's elements and defines the relationships between them. The model must have the ability to deal with several specific problems due to the general complexity of different elements: the hands, face, and feet in a human, or for some robot the differences between various probes, manipulators, and connectors. How these elements interact or are combined is of great importance. Whatever model is used, its final complexity will be highly related to the overall complexity of its total animation.

Because the relationship between the body's model and its associated motion exists, most modelings are designed to take into account several types of information that are not considered of a purely geometrical nature. These other forms include such things as physical properties like mass, density, and so forth, or mechanical constraints such as the maximum angle that a robot manipulator arm can bend. Adding this additional information to the geometric model increases the necessary computational time, but it makes for more realistic motions. So, as in most games you play with your computer, there is the usual tradeoff between realism and number crunching.

STICK MODELS

The simplest and most widely applied geometric model is the so-called "stick figure system." It consists of a hierarchical set of rigid objects (limbs) connected at joints (node points) which form an articulated body. The complexity of this model depends on the number of limbs and joints involved. Each node can have three degrees of freedom with the total model being as complex as necessary. This type of model is easily stored in your computer as a type of tree structure. Limbs are represented by the nodes and the joints are represented by arcs. In this form the modeling allows the definition of modular models formed by subtrees as "arm manipulator trees," "ambulatory trees," and so on. This subsetting of elements is useful for testing separate parts of the synthetic body. The main advantages of the stick body derive primarily from its ease of movement specification. It is only necessary to provide a single 3-D transformation matrix for each joint, corresponding to its three degrees of freedom.

Complex interconnected actions of composite structures have been successfully achieved using single planar or 3-D chains of links throughout a structure. The kinematic constraints associated with these chains are sufficient to determine movement without defining all the degrees of freedom. If you want to implement some of these stick-modeling techniques, I refer you to references 1-3. However, the major drawback to this type of model is that it produces unrealistic visualizations. The stick model's lack of volume makes depth perception difficult and consequently causes ambiguities in the animation sequence. Still, these stick models are widely used to define the motion of 3-D actors with their skeletal structure being covered after a movement with the necessary surface/volume imaging.

The skeletal synthetic figure shown in Figure 2 is defined as a set of segments corresponding to limbs and joints, with each joint being defined as the intersection of two segments. The angle between any two of these segments is called the joint angle or arc. It may have at most three types of position angles: flexion, pivot, and twisting. A flexion is a rotation of the limb which is influenced by the joint and causes the motion of all limbs linked to this joint. The flexion is carried out relative to the joint point and its axis must be defined. A pivot makes the flexion axis rotate around the limb which is influenced by the joint. Twisting causes a torsion of the limb which is influenced by the joint, with the direction of the twisting axis being found similarly to the direction of the pivot [12]. The actual positioning of newly calculated points using this skeletal technique is highly important and must be done with care. If a skeletal point is badly positioned, the joint will probably cause abnormal surface deformations after the mapping of surface shapes takes place. Points should therefore be positioned at the center of a joint with all points representing the extremity of the limb being at the center of the extremity of the limb [12].

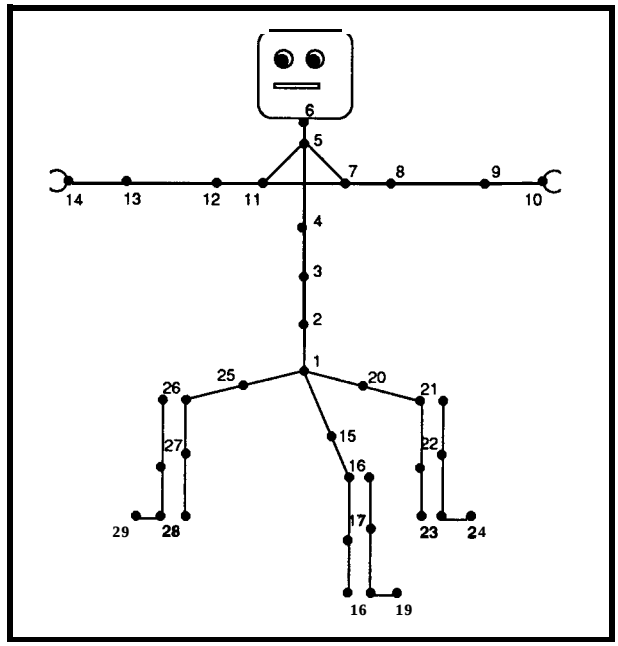


Figure 2—*The basic skeletal structure of a three-legged robot with flexible hip joints and two servo arms and head sensor section. Each of the three wheel mounts for the robot are represented twice in order to show the wheel orientation and direction.*

SURFACE MODELS

Unlike the stick model which represents a synthetic actor by its skeletal structure, the surface model is designed to simulate the external shape of the actor.

The primary difficulty with the surface model is that specification of movements is virtually impossible. Generally this problem is overcome by mixing models. A skeletal model is used to specify the movements and a surface is then modeled around the skeletal limbs in order to give them a more realistic volumetric shape. For more information

on how you can implement these techniques, I refer you to references 4–6.

VOLUME MODELS

Volume models are implemented by approximating the structure and shape of the synthetic actor with a collectionofelementalvolumeprimitives such as ellipsoids, spheres, and/or cylinders. Ingeneral thevolumemodel solves the problem of the inconsistent appearance of the stick models while providing an additional help to the overall animation rendering problem. For some practical applications, such as in collision detection [7] or choreography [8], they seem to be the best solution to modeling. As with the surface model, the volume model can also becombined witha skeletal model in order to facilitate the specification of movement [9,10].

GENERAL MOTION

Synthetic actor motion is implemented by defining a computer image with a set of parameters which describes the structure of the scene, its individual objects and their attributes, the position of the observer, and the position of all light sources and their attributes. The animation is then accomplished by varying the values of some or all of these parameters as a function of the sequence time. Since different components of the overall environment may undergo change at different moments, and they may 'behave" differently, they must be synchronized.

Most animators use one of three basic models for movement definition, either in a kinematic or a dynamic mode. Objects within the dynamic models must be defined using mechanical elements, such as the material (mass) and the joints (rods and springs with moments of inertia, etc.). Independent of the chosen computational mode, these three systems are usually classified according to the degree of movement that they allow. They are called the *guiding* (key frame and interpolation) model, the*program*-level (languages) model, or the tusk-*level* (motor program handling model).

The guiding systems model is a key frame procedure that defines a set of key frames and generates intermediate pictures between any two successive key frames by interpolation. This technique has been successfully extended to the use of a sequence of forces and torques applied to successive time spaced scenes.

Program-level systems are based on the use of some computer animation programming language which is typically the extension of some general-purpose language using a kinematic or dynamic model on a parallel processing system. If you would like to know more about these programming languages, I suggest you look into reference 11.

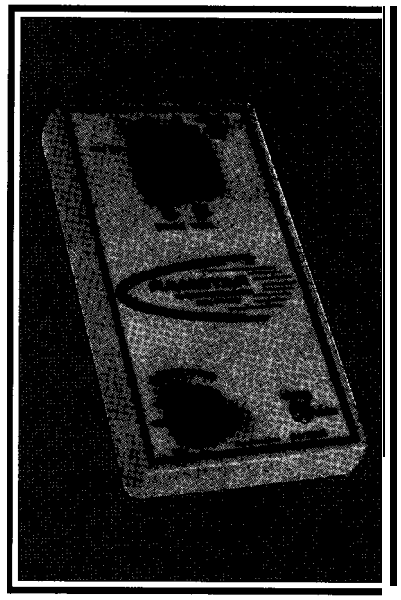
Task-level models are perhaps the most widely used. These models involve what is called motor program handling, where high-level commands perform predefined or computable movements. Here, once an externally applied action is specified, the **required** motion is computed according to the laws of motion chosen. In this form, the animation system must schedule the execution of the motor programs to control the synthetic actor with the motor programs themselves, generating the necessary action vectors controlling each element of the actor and the environment. To do this, a knowledge base of objects and figures within theenvironment isnecessary, containing information about their position, physical attributes, and functionality. In this technique, the animator can only specify the broad outlines of a particular action and then the animation system fillsin the details. As such, it is the closest we can come to the real world of robotics and the simulation of interactive behavior. It is a free-running system without user input after the initial setup profile. It must therefore have the ability to handle a wide variety of synthetic actor versus environment interactions.

RENDERING

This aspect of the animation problem involves the creation of a final product which is as realistic as possible. It's basically the application of

FAST TEST DRAM

64K - 128K - 256K
4X64K - 4X256K
1 Meg - 4Meg
DIPs - SIMMs- SIPs



RAMSTAR

Ins. RESOLUTION

ACCESS SPEED VERIFICATION
80 ns. thru 180 ns. (Base) $249.1
45 ns. thru 110 ns. (Fast) $349.(

AUTO-LOOP
Continuous Test 6.25 MBits/sec.

OPTIONS:

SIMM/SIP ADAPTER $189.
Tests 64K, 256K & 1 M Devices
8 or 9 Bit versions

NEW SIMSTAR ADAPTER $431.
Tests All Bits Simultaneously
64K, 256K & 1 M Devices
8 Bit, 9 Bit & PS/2 SIMMs or SIPs
Production Test Model $531.

4 X ADAPTER $ 89.
Tests 64K & 256K By 4 Bit Devices

AC ADAPTER $ 18.
Regulated +5V @ 1 Amp.

FREE RAMFACTS
DRAM NEWSLETTER

1-800-RAMSTAR

COMPUTERDOCTORS
9204-B Baltimore Boulevard
College Park, Maryland 20740
MADE IN U.S.A. PATENT PENDIN

Reader Service #114

correlated image synthesis techniques to each sequence frame within the animation. In general it involves such thingsashiddenline/surfaceremoval problems and shading, texture mapping, antialiasing, and so on. For more details, see "Image Synthesis: A Tutorial" in CIRCUIT CELLAR INK #12.

MECHANICS AND ROBOTICS

In order to use an animated sequence to predict and leamabout robot behavior, the animation has to be as representative of real-world motion as possible. In that sense, the best animation must be based on a detailed simulation which accounts for the dynamics of the action derived from a solid mathematical model. At the simplest level, this is accomplished by a kinematic approach based on specifying the position and orientation of an object at some time and then interpolating for intermediate times. In a more complicated form, goal-directed models based on dynamic physical systems have been constructed and designed to demonstrate more detailed behavior. Systems of this sort have been structured to execute a sequence of high-level commands. For example, suppose we design an animator which responds to high-level instructions like: "walk to the door and open it." It would then be up to our synthetic actor model to calculate how far it must walk, where to position itself and itsmanipulator foropening the door, and how it must move relative to the door as it is opened. The animationsystemwould thenbecalled upon to produce the kinematic description for each of the low-level actions that must take place to execute the sequence.

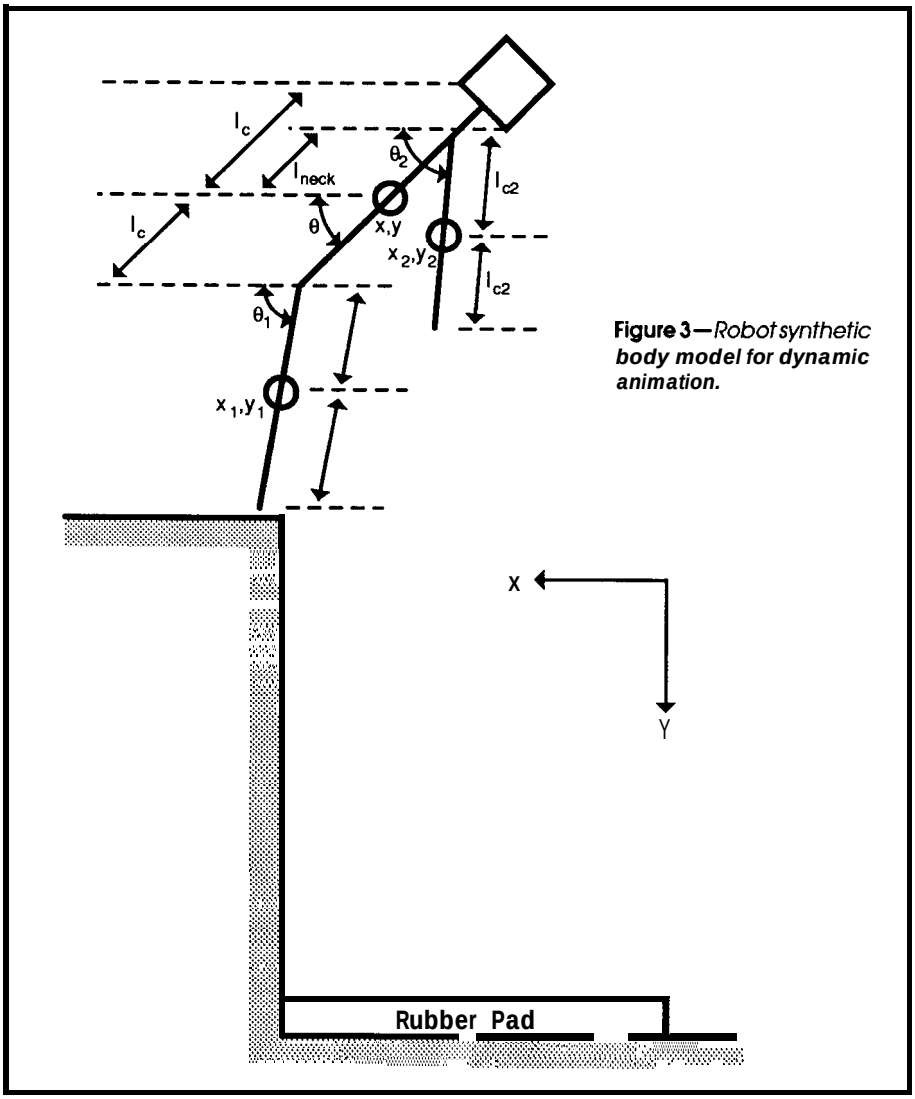


Figure 3—*Robot synthetic body model for dynamic animation.*

As it is, many different varieties of models of this sort have been created, and there has been an ongoing intensive research effort for goal-directed systems within the field of robotics. And, for our application, this has been especially true, since this type of model represents the link between pure animationand actual robot-world actions.

As I have previously stated, in order to create a detailed description of an animated sequence, the kinematic properties of position, velocity, and acceleration (for each point) can be calculated using a dynamic approach. The essence of this dynamic technique lies in the fact that each motion is described by a set of differential equations. The derivation of these equations can be accomplished two ways. The first method involves the use of basic algebra to combine simple descriptions of the forces (f=ma) and the torques (T=I@, where @ is the angular acceleration) together to form the required differential equations. This **turns** out to be a bit tedious, so I typically rely on my graduate physics "Classical Mechanics" textbook **by H.** Goldstein, and use the concepts of energy and Lagrange's equation.

In all honesty, I usually have difficulty deciding where to place the appropriate forces and torques. So, I always take the easy route and use Lagrange's equations since they make problem specification much easier. In thatform I need only deal withenergy as a scalar instead of a million complicated vector quantities, and I can also use easily written constraint equations to define the extent of any desired motion undertaken by my synthetic actor. To provide you with an example of how this can be done, let's consider our three-legged robot shown in Figure 2. Our ultimate goal will be to set up a series of generalized force equations based on a specified sequence of events that can be used to animate the robot. We will allow the robot to fall from of a high wall, strike a rubber mat, and then bounce.

To make this problem solvable (for demonstra tion purposes here), we must make a few simplifying assumptions. We will require that our robot

lie in a vertical plane, initially leaning over the edge of the wall. Its synthetic body will be constructed of three segments: the main body, an arm representing its two manipulators, and a leg segment representing its three-wheeled legs (see Figure 3). Each of these segments willbedefined in terms of the center-of-mass (CM) of its original multiple elements. This is done to enable reconstruction of a finite set of possible animation sequences demonstrating the motion of all the limbs, with all gross motion still defined by our Lagrangian solution. These sequences will not be unique, however, since the CM description is not unique forallorientationsof itsassortcdlimbs. But it does assist in the reconstruction phase. If the initial limb orientation is known, constraints on their motion can be used to predict their individual behavior.

In general, the Lagrangian (L) is defined in the follow manner,

$$L = T - U \qquad (1)$$

where T is the total kinetic energy and U is the total potential energy of the system. For generating differential equationsof motion, Lagrange'sequations take on the following form:

$$\frac{\delta L}{\delta q_i} - \frac{d}{dt}\left(\frac{\delta L}{\delta \dot{q}_i}\right) + \sum_k \lambda_k \frac{\delta f_k}{\delta q_i} = -Q_i \qquad (2)$$

where,

L = Lagrangian

q_i = i'th coordinate where $q_i = x, y, \theta, x_1, y_1, \theta_1, x_2, y_2, \theta_2$

f_k = k'th constraint equation where $f_k = f_{1x}, f_{1y}, f_{2x}, f_{2y}, f_{legx}, f_{legy}$

λk = k'th undetermined multiplier where $\lambda_k = \lambda_{1x}, \lambda_{1y}, \lambda_{2x}, \lambda_{2y}, \lambda_{legx}, \lambda_{legy}$

Q_i = generalized force or torque applied to the i'th coordinate

Here λ_k is the force that maintains the k'th constraint. For example, λ_{legx} is the x component of the tension in the leg between its CM and the point where the leg's wheel touches the top of the wall. The components of the generalized force are given by Q_i, a value which allows for the introduction of a damping force, an elastic bounce (experienced by the robot when it hits the rubber mat), and the torques that limit joint motion.

The following is a brief description of the major stepsrequired in order to apply Equation 2 and determine a set of generalized force equations defining our animation sequence:

Define our coordinates.

Figure 3 displays our choice of body model used in this animation. Since the right and left sides of the body are coupled through the CM procedure, we need only nine coordinates to describe the position of the body. These are,

x,y = the position of the CM of the body (of mass = m)

x_1,y_1 = the position of the CM of the three-wheeled legs (of mass ml)

x_2,y_2 = the position of the CM of the two-arm manipulators (of mass m2)

θ = the angle between the body and the horizontal

θ_1 = the angle between the legs and the horizontal

θ_2 = the angle between the arms and the horizontal

Write theconstraint relations between the coordinates.

There exist within this problem two systems of differential equations based on two different constraint conditions. The first condition relates to the position of our robot while it is on the top of the wall, just before it falls. At this point, the wheeled leg is constrained to the platform, such that:

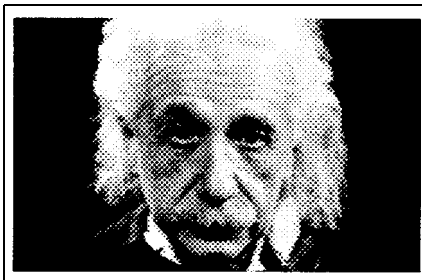
$$f_{legx} = \text{leg}, -x_1 - l_{c1}\cos\theta_1 \qquad (3)$$
$$f_{legy} = \text{leg}_y - y_1 - l_{c1}\sin\theta_1 \qquad (4)$$

The second set of constraint equations have to do with the motion of the joints. These have the following form:

$$f_{1x} = x_1 - x - l_c\cos\theta - l_{c1}\cos\theta_1 \qquad (5)$$
$$f_{1y} = y_1 - y - l_c\sin\theta - l_{c1}\sin\theta_1 \qquad (6)$$
$$f_{2x} = x_2 - x - l_{neck}\cos\theta - l_{c2}\cos\theta_2 \qquad (7)$$
$$f_{2y} = y_2 - y - l_{neck}\sin\theta - l_{c2}\cos\theta_2 \qquad (8)$$

Write the kinetic and potential energy.

The total kinetic energy of our synthetic actor can be written as a sum



Introducing VICTOR, the video capture and image processing library

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

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

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

Victor and Zip support ImageWise and other popular video digitizers.

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

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

VICTOR LIBRARY includes FREE ZIP Image Processing $195

VICTOR LIBRARY with video frame grabber $349

ZIP Colorkit $75

Call (314) 962-7833 to order

VISA/MC/COD

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

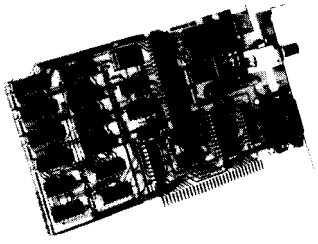
eader Service XI 10

INTRODUCING
on-axis™
4 CHANNEL 24-bit
QUADRATUDE-MODE COUNTER

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

$299.00 INTRODUCTORY PRICE
(add $150 for optional connector to Bausch & Lomb "glass scales".)

étude™
25 MHz 8-bit
ANALOG-TO-DIGITAL CONVERTER



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

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

PRICE: $495.00

Each product requires: PC compatible ½ length 8-bit expansion slot. DOS 2.11 or greater. EGA, VGA or Hercules display needed for graphic representation of data.

Silicon Alley Inc.
P.O. BOX 59593
RENTON, WA 98058
(206)255-7410

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

Reader Service #159

of the kinetic energy (relative to the CM) and rotational energy of each segment:

$$T = \frac{1}{2}mv^2 + \frac{1}{2}Iw^2$$

where,

v = speed of the CM

w = angular velocity of the body about the CM

$$v = \sqrt{(\dot{x}^2+\dot{y}^2)}$$

($\dot{x},\dot{y}$ are the components of the velocity vector)

$$w = \dot{\theta}$$

so,

$$T = \frac{1}{2}(m(\dot{x}^2 + \dot{y}^2) + m_1(\dot{x}_1^2 + \dot{y}_1^2) + m_2(\dot{x}_2^2 + \dot{y}_2^2)) + \frac{1}{2}(I\dot{\theta}^2 + I_1\dot{\theta}_1^2 + I_2\dot{\theta}_2^2) \quad (9)$$

Here, the moments of inertia for the arms, wheeled legs, and body have the form,

$$I = \frac{1}{12}ml_c^2 \quad (10)$$

$$I_1 = \frac{1}{12}m_1l_{c1}^2$$

$$I_2 = \frac{1}{12}m_2l_{c2}^2$$

The potential energy has the form $U=mgy$ wherey is the height of the *CM* above a specified reference level. For our problem it is given by Equation 11.

$$U = g(my + m_1y_1 + m_2y_2) \quad (11)$$

Use *Lagrange's equation to define the differential equations.*

Applying Equations 1 and 2 to the total Lagrangian will produce a set of differential equations which have to be solved for the various accelerations. These required accelerations are:

$$\ddot{x}, \ddot{y}, \ddot{\theta}, \ddot{x}_1, \ddot{y}_1, \ddot{\theta}_1, \ddot{x}_2, \ddot{y}_2, \ddot{\theta}_2$$

Differentiate the constraint equations to get enough equations so that all the accelerations can be solved for.

Using the above results, a linear system of nine equations for 15 unknowns can be created. These 15 unknowns consist of the above accelerations and the following Lagrange multipliers:

$$\lambda_{1x}, \lambda_{1y}, \lambda_{2x}, \lambda_{2y}, \lambda_{legx}, \lambda_{legy}$$

Each of the accelerations can be found by taking the second time derivatives of the constraint equations shown in Equations 3-8.

Write equations for fhegeneralized forces, the limits on limb motion, the damping of motion,and environment interaction constraints.

Each joint is structured to have an equilibrium position with negative and positive limits defining how the torque increases away from the equilibrium condition. These torques act as a restoring torque, limiting the absolute motion of the joint. Our system is also designed to have two built-in damping forces to limit linear and rotational motion of the form,

$$F_{damping} = -b\dot{x} \quad (12)$$

and

$$T_{damping} = -b\dot{\theta}$$

where b is the damping constant (>0).

Since our robot is going to bounce off a rubber mat, we need also to define a spring force acting on the CM of our robot which will push it upward in a vertical direction. A simple form of this action can be described by,

$$F_{rubber} = 0 \quad \text{if } y > y_{rubber} \quad (13)$$
$$= -K(y - y_{rubber}) \text{ if } y \le y_{rubber}$$

where *K* is the spring constant (K>0), and y_{rubber} is the vertical position of the mat. We can now combine all of our proceduresdescribedaboveand write down the generalized force equations for our animation sequence:

$$Q_x = -b_x\dot{x}$$

$$Q_y = -b_y\dot{y} + [0 \text{ if } y > y_{rubber} \text{ or } -K_y(y - y_{rubber}) \text{ otherwise}]$$

$$Q_\theta = -b_\theta\dot{\theta}$$

$$Q_{x1} = -b_{x1}\dot{x}_1$$

$$Q_{y1} = -b_{y1}\dot{y}_1 + [0 \text{ if } y_1 > y_{rubber} \text{ Or-K}_{y1}(y_1 - y_{rubber}) \text{ otherwise}]$$

$$Q_{\theta 1} = -b_{\theta 1}\dot{\theta}_1 + T_{restore\theta 1}(\theta_1 - \theta)$$

$$Q_{x2} = -b_{x2}\dot{x}_2$$

$$Q_{y2} = -b_{y2}\dot{y}_2 + [0 \text{ if } y_2 > y_{rubber} \text{ or } -K_{y2}(y_2 - y_{rubber}) \text{ otherwise}]$$

$$Q_{\theta 2} = -b_{\theta 2}\dot{\theta}_2 + T_{restore\,\theta 2}(\theta_2 - \theta)$$

where $\theta_1 - \theta$ is the relative angle between the leg and body.

In the above example, we simplified the problem by limiting our robot to five degrees of freedom. As a result we ended up with nine differential equations. A more complete synthetic actor (in 3-D) would have produced 30 to 40 degrees of freedom. However, since one of the primary uses of the dynamic method is the prediction of subsequent behavior resulting from a set of applied forces, it is a good tool for simulation of real-world robotic situations.

TASK PLANNING AND EXECUTION OF INTELLIGENT ACTIVITY

As in a robot task-level system, actions within a task-level animation model are specified only by their actions within their environments. In each, the ultimate goal is to plan the operation of the robot or synthetic actor in such a way that they are able to complete a set of specified motions within their respective environments. For example, a few typical tasks undertaken by a synthetic actor could be: moving from one point to another; picking up an object at one location and moving it to another; learning a mode of operation (finding a suitable path from one point to another and learning it using AI techniques).

Each requires the development of a low-level set of instructions for sequence completion. The actual procedure for generating these low-level task instructions hasbeendivided into three phases. They are called world modeling, task specification, and manipulator program synthesis. The following is a brief description of each.

WORLD MODELING

World modeling consists mainly of constructing the gross characteristics of the animation model and the background scene. This involves defining the geometry and physical elements of the synthetic actors, the scene, and all objects, with all constraints on their motions being defined. These constraints obviously depend on shape of the objects and actors as well as their respective geometric positions. The most common way these interrelationships are modeled is through the use of a facet-based representation, CGS [13], or a soft object model [14]. What is important to keep in mind is that most synthetic actor motion generates conditions of deformed bodies; it is therefore necessary to incorporate corrections for this effect into your model.

TASK SPECIFICATION

There are many ways to specify tasks within a task-level system. It can be achieved **by example,** by a sequence of model states, or though a sequence of commands. Defining a task by example represents a learning procedure

Total control with LMI FORTH™

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

For Development:
Interactive Forth-83 Interpreter/Compilers for MS-DOS, OS/2, and the 80386
- 16-bit and 32-bit implementations
- Full screen editor and assembler
- Uses standard operating system files
- 500 page manual written in plain English
- Support for graphics, floating point, native code generation

For Applications: Forth-83 Metacompiler
- Unique table-driven multi-pass Forth compiler
- Compiles compact ROMable or disk-based applications
- Excellent error handling
- Produces headerless code, compiles from intermediate states, and performs conditional compilation
- Cross-compiles to 8080, Z-80, 8086, 68000, 6502, 8051, 8096, 1802, 6303, 6809, 68HC11,34010, V25, RTX-2000
- No license fee or royalty for compiled applications

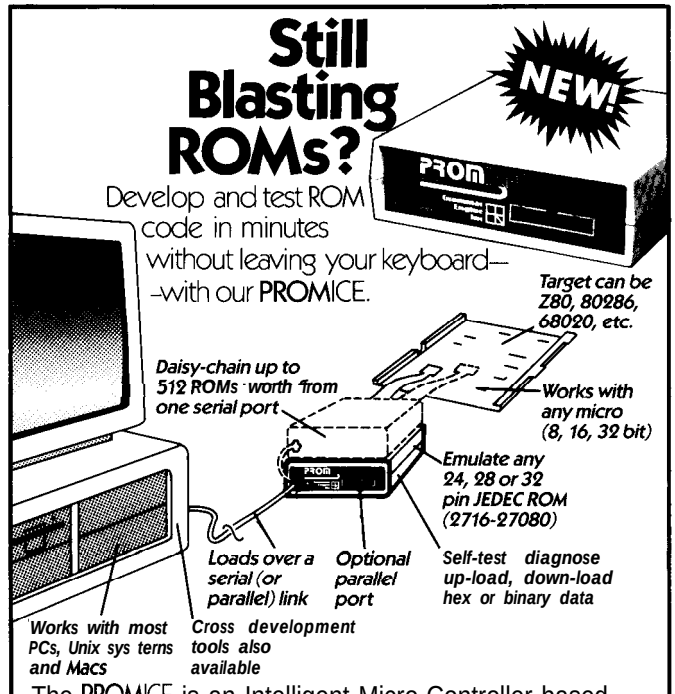
Laboratory Microsystems Incorporated
Post Office Box 10430, Marina del Rey, CA 90295
Phone Credit Card Orders to: (213) 306-7412
FAX: (213) 301-0761

Reader Service x137

Still Blasting ROMs?

NEW!

Develop and test ROM code in minutes without leaving your keyboard— —with our PROMICE.



The PROMICE is an Intelligent Micro-Controller based unit having optional Analysis Interface for Firmware Debugging Support. For more information, call or write:

Grammar Engine, Inc.
3314 Morse Road
Columbus, Ohio 43231
6141471-1113

VISA and Mastercard Accepted
Dealer Inquiries Welcome

In Pacific and Mountain time zones
415/595-2252

Reader Service #128

where the animator performs the task at least once in order to explain it to the system. Model state implementation is a bit more realistic. Sequences of key frames can be constructed based on the set of relations defined within theattribute table so that intermediate frames based on parametric changes can be interpolated later. However, the best implementation is achieved by developing a truly high-level command structure which specifies the broad outlines of a particular movement with the animation system filling in the details.

MANIPULATOR PROGRAM SYNTHESIS

Inrobotics theoutput of this phase is typically a program in a manipulator-level language. In other words, a sequence of timed control signals sent to different equipment to instigate necessary actions. In the computer animation world, however, it's slightly different. Here, several different types of output are possible. We can obtain a completed animation sequence composed of a series of frames ready for recording, the values of parameters for parametric key frame interpolation evaluation, or we could generate a script designed to drive an animation language like ASAS [11] or CINEMIRA [15]. In effect, we are just choosing the output form of our animation system compiler. The primary difference between this stage and the task-planning phase is that the task-planning step defines and sets up the problem while the program synthesis process creates the actual animation sequence.

TRAJECTORY PLANNING AND OBSTACLE AVOIDANCE

Trajectory planning is the process of converting a task description into a planned path or sequence of positions, velocities, and accelerations (x,y,v,a). For synthetic actors this motion is typically described by joint coordinates. Of course the actual coordinates used depends on the type of physical model specified. For key-frame systems, this means defining (x,y,v,a) at selected times and then interpolating in between. For automatic trajectory planning (as in a kinematic or dynamic model) this represents defining the start and goal positions and allowing the system to determine the optimum trajectory, taking into account all potential obstacles.

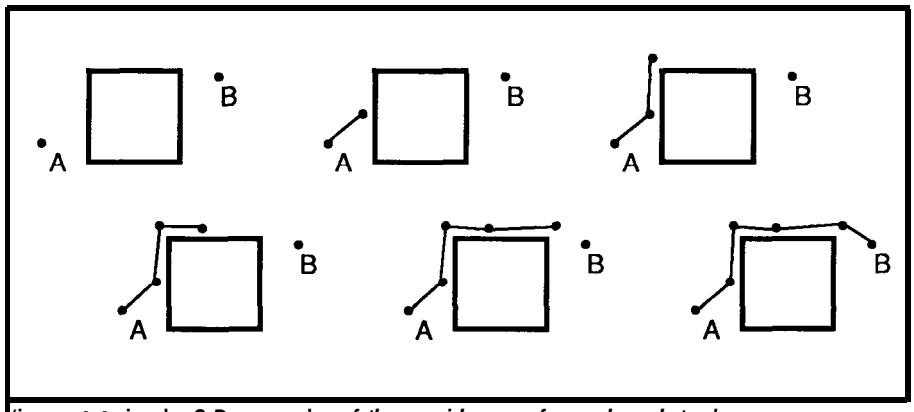


Figure 4-A simple 2-D example of the avoidance of a cube obstacle.

The central aspect to trajectory planning is the avoidanceof collisions with obstacles in the planned path. In robotics this is the standard geometric problem of finding a path for a moving solid among other solid obstacles, called the "findpath" problem. Whether for robotics or animation, the simplest solution to the problem is trying a path and testing it for potential collisions. If collisionsaredetected, a new path is then chosen. A practical form of such a findpath algorithm can be created by the repetition of three basic steps. First calculate the volume swept out by the moving object along the proposed path, then determine the overlap between the swept volume and the obstacle, and finally propose a new path for avoidance using previous knowledge.

To demonstrate this technique, consider the outline of a simple findpath algorithm in Listing 1 [16]. It is designed to determine the trajectory (without collision) of a moving object 0 from point A to B. The environment, D, is composed of variety of static objects F,, modeled using facet-based surfaces.

A more detailed description of how this "simple" obstacle avoidance techniqueisimplemented isdescribed in Thalmann's book *Animating Synthefic Actors* Using *Artificial Intelligence* and *Robotics* [16]. Of course the obstacles within this example are not moving. But, what happens if they do?

Obstacles are not necessarily fixed in space. They can be moving around during the animation of the principle syntheticactor.Or,moreinterestingly, the obstacles themselves may have individual behavioral characteristics; and under certain circumstances, our synthetic actor may be called upon to navigate that dynamic environment without colliding with these objects. Like any difficult problem in modeling, there are as many solutions to this problem as there are inventive problem solvers. But, since I'm the typical hard-headed physicist, I always love the approach that uses some basic concept in physics or is directly and beautifully simple in character. I have therefore found two basic techniques, thatparticularlycatchmyfancy.These are the force field concept and the steer-to-avoid method. In general, the force field method works in relatively undemanding situations where relative motion between objects is on a reasonable time scale. The steer-to-avoid technique is much more robust and seems closer in spirit to the actual naturalmechanism.Itsonlydrawback is that the time interval between synthesized frames must be small for high-speed motion.

The force field model [17] postulates a field of repulsive force emanating from the obstacle out into space, with the motion of the principle object being increasingly repulsed as it gets closer to the obstacle. It is a simple

```
Simplify each Fi
!   each object is simplified as a collection of parallelepipeds
    parallel to the axis
Find the center and the size of O
!   calculate the volume swept out by the moving object 0
    along the proposed path and determine the overlap
    between the swept volume and the obstacle
while O cannot reach B do avoid the obstacle
Advance to B
Simplify the trajectory

boolean function VISIBLE(O)
VISIBLE:=TRUE
create a line d passing through A and B
for each Fi
    for each facet fki of Fi
        if fki is not parallel to d then
            create a plane P from fki
            find the point Ik=P d
            if I is inside fki then
                if I is between A and B then
                    VISIBLE:=FALSE
                    store  I,  store the distance to A,
                    store the normal to fki and fki
            if not VISIBLE then
                return the nearest intersection from A

boolean function OBSTACLE
create a BOX B around O
OBSTACLE:=FALSE
for each vertex of B
    SEE (VISIBLE)
    OBSTACLE:=OBSTACLE or (not VISIBLE)
    if not VISIBLE then
        store information about contact
    if not OBSTACLE then
        retrieve the nearest facet f from the starting point
        calculate the position K of the center in order to have
          0 at a distance e of the obstacle
        return f and K

AVOIDANCE

ADVANCE to the desired contact
!   advance to a point P consist of adding P to the trajectory.
    The procedure is only called when the moving object is in
    front of a corner of the parallelepiped; it tests which
    coordinate does not vary and adds to this coordinate the
    width of the box B.

    if we have just turned then
        find a point to go towards the same direction
    else
        find the next visible point or a point to go towards the
          same direction
    if OBSTACLE then
        TURN to avoid the obstacle without specifying the
          direction
    else
        ADVANCE to the determined point
        calculate the next point to finish to avoid the facet
          using the procedure TURN
        if VISIBLE (P) then
            ADVANCE to P
        else
            turn to avoid the new obstacle using the same
              direction
```

Listing 1 — *This simple findpath algorithm is designed to determine the trajectory of a moving object 0 from point A to point B.*

model to create: the geometry of the field is usually defined as a $1/r^2$ repulsive force such that an avoidance acceleration can be directly calculated from the field equation. If the principle object approaches an obstacle surrounded by a force field, at an angle such that it is exactly opposite to the direction of the force field, the object will not turn away. In this case the force field servesonly to slow theobject down by accelerating it backwards with no sideways motion. As a result the worst reaction to a collision is to fail and turn back. On the other hand, if the object approaches a wall, the force field method isdesigned to cause it to turn away. However, special conditions must be included within the model to enable the principle object to ignore the wall if its motion is parallel to it. The primary difficulty with this approach is that the force fields tend to be too strong up close and too weak far away. But, this can be overcome by carefully modeling the force field to have specific distance functionality and using short time interval steps within the sequence coupled to wide "peripheral vision." Application within static systems are very effective since they involve the calculation of a path based on minimum repulsive interference.

The steer-to-avoid is perhaps the most natural form of simulation. The principle object considers only objects directly in front of it. Within its perspective local space, it finds the silhouette edge of the obstacle closest to the point of eventual contact. A radial vector is then computed which will aim the object at a point one body length beyond the silhouette edge for avoidance, with these changes in direction being weighted by the original start to goal direction.

Again the actual choice of an obstacle-avoidance technique depends highly on the form of your basic animationmodel.Useof theforcefield technique is highly successful while employing a dynamic model simulator. Each obstacle's force field equations and implied constraints couple nicely with the equations of the motion of the principle actor. Your limitation lies only in computational capacity. If you employ a parametric key-frame procedure, the steer-to-avoid also works effectively. Here, the calculation of avoidance vectors are easily accomplished at each key-frame interval, provided these intervals are sufficiently small to allow for necessary directional changes. But one thing you must keep in mind: most of these algorithms developed for obstacle avoidance have direct application within the robotics environment. If your robot employs a sensor system that enables it to locate and evaluate the motion and spatial displacement of obstacles within its field of activity,

then these same algorithms can be used to generate avoidance instructions for the robot's guidance control center.

AND ONWARD

Well, I've run out of space and words. But I hope I've given you a little insight into the worlds of robotics, animation, and AI. In my mind, they are all the same, irrecoverably intertwined. So have fun and don't think of yourself left out because you can't afford a fancy robot and control system. Your PC can create your robotic world for you. Give it a little AI and see how it behaves. ✣

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

IRS

207 Very Useful
208 Moderately Useful
209 Not Useful

REFERENCES

1. Calvert T., J. P. Chapman. "Aspects of the Kinematic Simulation of the Human Movement," IEEE Computer Graphics Application (November 1982).

2. Girard M., A. Maciejewski, 'Computational Modeling for the Computer Animation of Legged Figures,' Proc Siggraph 1985, v 9, no 3.

3. Korein V., N. Badler, "Techniques for Generating the Goal-Directed Motions of Articulated Pictures," IEEE Computer Graphics Application (November 1982).

4. Armstrong W., M. Green and M. Lake. "Near Real Time Control of Human Figure Models.' and "Dynamics for Animation of Characters with Deformable surface,' Graphics Interface 1986.

5. Sadler N., S. Smoliir. 'Digital representations of Human Movement,' ACM Computer Surv, vol 11, (March 1979).

6. Calmull R., "Computer Display of Curved Surfaces," Proc IEEE Conf Compu Graph Appl (November 1982).

7. Badler N., J. O'Rourke and H. Toltzis, "A Spherical Representation of a Human Body for Visualizing Movement.' Proc IEEE. vol 67, no 10, (October 1979).

8. Herbison-Evans D., "Real-Time Animation of Human Figure Drawings with Hidden Lines Omitted," IEEE Compu Graph Appl (November 1982).

9. Tost D., X. Pueyo, "Human Body Animation: A Survey, Siggraph '88 Course Notes, vol 4, pp 12–22.

10. Badler N., J. O'Rourke and b. Kaufman, "Special Problems in Human Movement Simulation," Proc Siggraph 80.

11. Reynolds C., "Computer Animation with Scripts and Actors," Siggraph 1985, Tutorial Notes.

12. Magnenat-Thalmann N. and D. Thalmann, "Construction and Animation of a Synthetic Actress," Siggraph 88 Course Notes, vol 4, pp 37–51.

13. Ciarcia C., "Image Synthesis: A Tutorial," Circuit Cellar INK, Issue 12.

14. Wyvill B., C. McPheeters and G. Wyvill, "Animating Soft Objects," The Visible Computer, 2(4):235–234, February 1986.

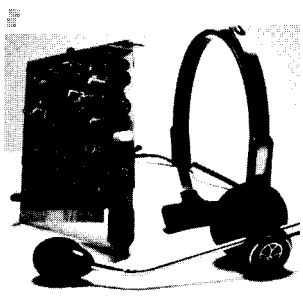
15. Magnenat-Thalmann N. and D. Thalmann, "The Use of High-Level Graphical Types in the MIRA Animation System," IEEE Computer Graphics and Applications, vol 3, no 9, Dec 1983, pp 9–16.

16. Magnenat-Thalmann N. and D. Thalmann, "Animating Synthetic Actors Using Artificial Intelligence and Robotics,' Prentice Hall, Englewood Cliffs, New Jersey, 1988.

17. Amkraut S., M. Girard and G. Karl, "Motion Studies for a Work in Progress Entitled EURYTHMY," in Siggraph Video Review, issue 21 (second item. time code 3:58 to 7:35), 1985, produced at Computer Graphics Research Group, Ohio State U, Columbus, Ohio.

TALK TO YOUR COMPUTER
WITH VOICE MASTER KEY® FOR PCs/COMPATIBLES
VOICE RECOGNITION WITH SPEECH RESPONSE

GIVE A NEW DIMENSION TO PERSONAL COMPUTING The amazing Voice Master Key System adds voice recognition to just about any program or application Voice command up to 256 keyboard macros from within CAD, DTP, word processing spread sheet, or game programs. Fully TSR and occupies less than 64K. Instan response time and high recognition accuracy. A real productivity enhancer!



SPEECH RECORDING SOFTWARE Digitally record your own speech, sound or music to put into your own software programs. Software provides sampling rate variations, graphics-based editing, anc data compression utilities. Create software sound files, voice memos, more. Senc voice mail through LANs or modem. A superior speech/sound development tool.

INTERACTIVE SPEECH INPUT/OUTPUT Tag your own digitized speech files tc voice recognition macros. Provides speech response to your spoken commands -- al from within your application software! Ideal for business, presentation. education, or entertainment programs you currently use. Augment the system for wireless uses in robotics, factory process controls, home automation, new products, etc. Voice Master Key System does it all!

EVERYTHING INCLUDED Voice Master Key System consists of a plug-in card, durable lightweight microphone headset, software, and manual. Card fits any available slot. External ports consist of mic inputs and volume controlled output sockets. High quality throughout. easy and fun to use.

ONLY $149.95 COMPLETE

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

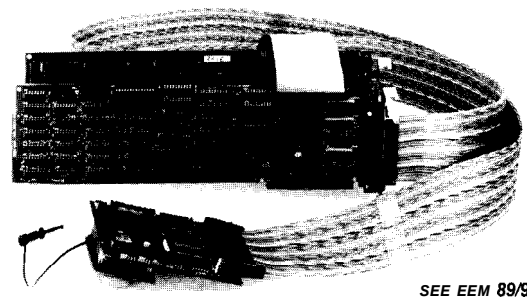
30 DAY MONEY BACK GUARANTEE IF NOT COMPLETELY SATISFIED.

CALL OR WRITE FOR FREE PRODUCT CATALOG.

COVOX INC.
675 CONGER ST.
EUGENE, OR 97402
TEL: (503) 342-I 271
FAX: (503) 342-I 283

Reader Service #117

68HC11
PC-based emulator for 68HC11



SEE EEM 89/90
Pages D 1324-1328

- PC plug-in or RS-232 box.
- Pull-down menus with full window support, combined with command-driven User Interface.
- Up to 3.3 MHz (E clock) real time emulation.
- No intrusions to the 68HC11's resources.
- 48 bit wide 16K deep trace. All functions usable without disturbing emulation. Time stamping. Two level trigger.
- Symbolic and C Source Level Debugging, including in-line assembler and disassembler.
- Supports A, E, D and F parts.

Prices: 64K Emulator and pod $2590: 4K Trace $1995'

CALL OR WRITE FOR FREE DEMO D/SK!

nohau CORPORATION
51 E. Campbell Avenue
Campbell, CA 95008
FAX (408) 378-7869
(408) 866-1820
'US only

Reader Service #148

FEATURE ARTICLE

David Otten

Part 1

Building MITEE Mouse III

The Hardware for a Maze-Running Rodent

MITEE Mouse (Massachusetts Institute of Technology Electrical Engineering Mouse) is a response to a robot contest first proposed in 1977 by the editor of *IEEE Spectrum*. After hearing of The Great Clock Climbing Contest proposed by *Machine Design*, which turned out to be a mechanical contest, Don Christiansen and his staff at *Spectrum* wanted to create a contest for electrical engineers, one that would involve the recently available microprocessor. They came up with the Amazing MicroMouse Maze Contest in which a robot had to find its way through a 10' by 10' maze without any external assistance. The contest was conceived in 1977 and run in 1979 at the National Computer Conference in New York. It was a great media event, but to the disappointment of many, a high-speed "dumb" wall follower fared better than many of the "brighter" mice. The contest was only intended to be run once, so micromouse competition was virtually dead in the United States until 1985.

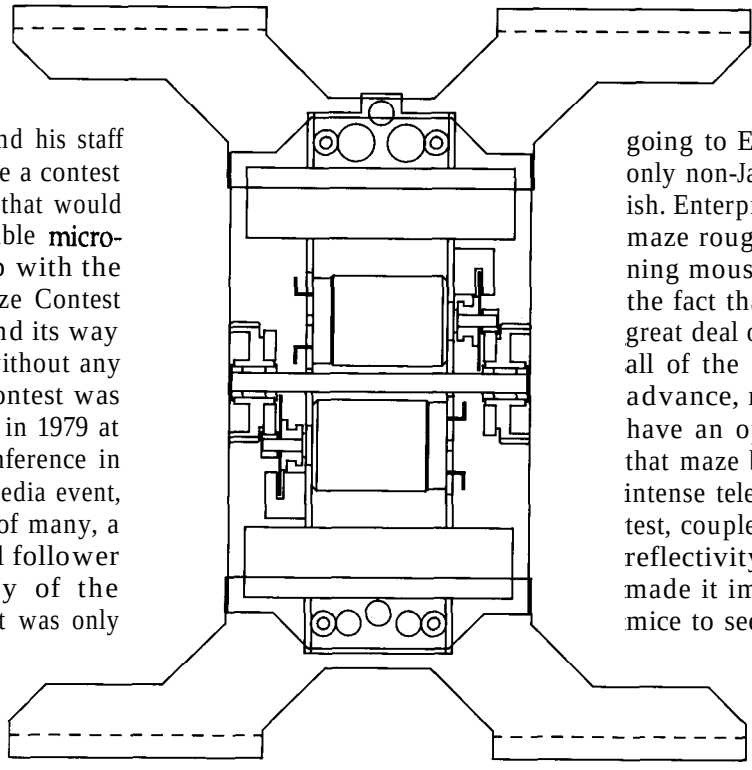


Figure 1 -*A top view of MITEE Mouse III shows the sensors on the four corners and the motor assemblies in the center.*

Meanwhile, Professor John Billingsley from Portsmouth Polytechnic in England made several minor, but critical, changes to the rules and introduced the contest at a conference called Euromicro held in London in 1980. The rule changes involved moving the goal from an edge of the maze to the center. With proper design of the maze, it was now possible to prevent a simple wall follower from getting to the center of the maze; some sort of intelligence was required. Several members of the Japan Science Foundation took the rules back to Japan and organized the first Japanese contest the same year. A Japan MicroMouse Association was formed, and they have sponsored contests in Japan every year since then. In Europe the contest was carried by Euromicro through 1985 and was then picked up by the IEEE.

In 1985 the Japanese, wishing to sponsor a world contest, sent mazes to a number of countries around the world, hoping that everyone would have an opportunity to practice on identical mazes. They also sent representatives to observe the national contests of those countries and paid for the winners to participate in the world contest. At that contest, held in Tsukuba City, Japan, the top six places were taken by the Japanese with seventh place going to Enterprise of England, the only non-Japanese mouse to even finish. Enterprise had a time through the maze roughly twice that of the winning mouse. It seems that in spite of the fact that the Japanese went to a great deal of trouble to send mazes to all of the participating countries in advance, most of the mice did not have an opportunity to practice on that maze before going to Japan. The intense television lighting at the contest, coupled with the relatively poor reflectivity of the top of the walls, made it impossible for many of the mice to see where they were going, and they weren't able to reach the center.

Thinking that the contest was dead after the 1979 event in New York City, we did not realize until just before the world event, that the contest was alive and well in Japan and Europe. Though it was too late to enter the world contest, we decided that if there was another contest in the United States in 1986, we would organize and field an entry. In October of 1985 we found out from Susan Rosenbaum, one of the organizers of the original event, and affectionately know as MicroMom for her association with

thecontest since then, that there would be a contest in Atlantic City in March of 1986. Our entry, MITEE Mouse I, came in dead last at that contest, but by the summer of that year we had improved it to the point where it came in second in London. MITEE Mouse II, our second attempt, was able to capture first place at both the national contest in Chicago in 1987 and the London contest a month later. Not until 1988 were we able to find out about a contest in Japan before it was actually held. MITEE Mouse III, the subject of this article, was designed to compete against the Japanese, and in November of 1988, it was finally able to do that at their 9th national contest held in Kawasaki City. We came in a close third, less than a quarter of a second behind second place, and less than a half second out of first place.

AN OPERATIONAL MOUSE

The basic goal of a micromouse is to travel from the start square of a maze to the finish square in the least amount of time. At the beginning of the contest the mouse knows nothing about any of the walls in the maze. Sensors on the mouse, however, are able to identify the status of the walls surroundingthesquare that themouse iscurrently in. It can therefore find out where the openings are and proceed to at least the next square. When there is more than one choice, the mouse must make a decision as to which way to go. To do this themouse keeps track of all the walls in the maze. Walls that it has never seen before are marked as unknown. Walls surrounding squares the mouse has visited are marked open or closed as appropriate.

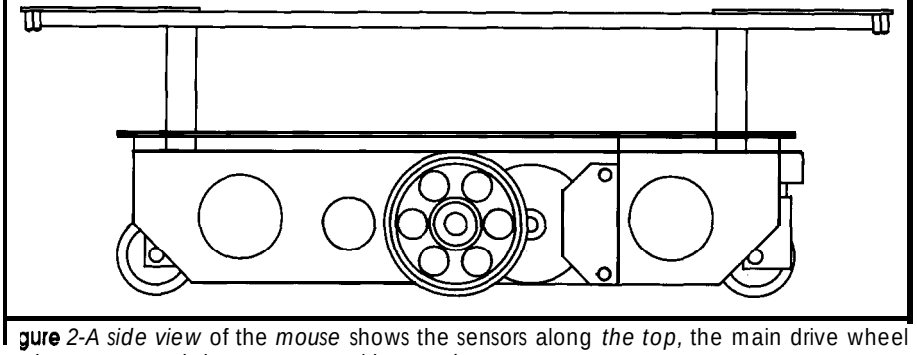
The mouse is very optimistic in its view of the maze. When it must make a decision as to which path to take, it solves the maze assuming that any unknown walls are open. It then starts out on the path prescribed by this solution until it comes to a square with unknown walls. In visiting that square the sensors pick up the true status of the walls and the maze map is updated. Since the true status may violate the previous assumption that the wall was open (made when it was unknown) the maze solution is again determined with this new information and the mouse proceeds to the next square. At the beginning of the contest when little is known about the maze, the mouse will stop in every square. As the contest progresses however, more is learned and it stops less often. At some point it is able to plot what it considers to be the best course from the beginning to the end without going through any squares with unknown walls. This, then, marks the end of the search phase for the mouse; it goes back to the beginning, and runs to the middle using the best path. The first run along the best path is usually done at a low acceleration to ensure that the mouse will not crash, and then the acceleration is increased with each successive run until it crashes. The run before the crash is presumably the best possible run (i.e., the fastest).

THE MECHANICAL MOUSE

Because MITEE Mouse III is a robot that physically moves through a maze, it must have a means of propulsion, as well as systems for navigation and sensing. Figures 1 and 2 show top and side views of the mouse. It uses a "wheel chair" configuration with two drive wheels, one on either side of the chassis. Each drive wheel is controlled by its own DC motor to permit acceleration, deceleration, and steering of the mouse. A shaft encoder is connected to each motor to permit accurate control of the motor position and velocity at all times. Casters in the front and back of the mouse absorb the reaction torque of the motors during acceleration and deceleration. Arrays of infrared sensors are located above the walls. These are arranged to see the tops of the walls which are painted a reflective red and to ignore the floor, which is painted a nonreflective black.

The goal of the mouse is to travel through the maze in the shortest time, once the maze has been solved. To minimize this time, the mouse travels with a constant force on the wheels and tires at all times. On a straightaway the force is used to accelerate for half the length of the straight and decelerate for the other half. During a turn the force is used to counter the centrifugal force on the mouse. Initial tests done with several different tire materials in connection with MITEE Mouse I indicated that the tires should be good for an acceleration of 0.89g (the force the tirescould exert was0.89 times the weight of the mouse). Based on this, the motors and batteries were sized to provide lg acceleration. This turned out to be much too large and MITEE Mouse I always skidded when theaccelerationwasgreater than0.25g. On MITEE Mouse III therefore, the motors and batteries were only sized for 0.5g operation. To date, MITEE Mouse III has only achieved 0.4g acceleration, so this is plenty.

The chassis is designed to provide a light, narrow mouse with a low center of gravity and a low moment of inertia. If the mouse is light it requires less force to accelerate and decelerate, resulting in smaller motors and batteries. A narrow mouse allows more margin for error when navigating through the maze. In addition, many maze designs include sections which are like a stair step, a left turn followed by a right turn, followed by a left turn,



gure 2-A side view of the mouse shows the sensors along the top, the main drive wheel the center, and the casters on either end.

and so on. If a mouse is narrow enough, it can go directly down this diagonal, thereby saving a great deal of time. The nominal width of the passage on a diagonal is 4.34 inches. MITEE Mouse III has a chassis width of 3 inches allowing approximately 0.67 inches of clearance on each side.

A low center of gravity keeps more of the weight of themouseon the main drive wheels and less on the casters during acceleration and deceleration. This allows the drive wheels to exert a greater force. The center of gravity for MITEE Mouse III is located directly above the drive wheels about 1 inch off the floor. Even at the maximum possible acceleration allowed by the tires (0.89g), 75% of the weight of the mouse will be on the drive wheels and only 25% on the front or rear caster.

In addition to a low center of gravity, a low moment of inertia is also desirable. This allows the mouse to turn quickly, resulting in the maximum speed through turns. To minimize the moment of inertia, all parts of the mouse should be as close as possible to an imaginary vertical line through the center of the mouse. The moment of inertia is proportional to the weight of a component multiplied by the square of the distance from that component to the imaginary line. Clearly heavy items such as the motors and batteries should be close to the center and the design of the mouse reflects that. Lightitems, however, that are far from the center, can also contribute significantly to the moment. In the case of MITEE Mouse III, the sensors which are located at the farthest extremities of the mouse contribute only 10% to the weight of the mouse but 26% to the moment of inertia. Figure 3 shows the weight distribution for MITEE Mouse III. The total weight is 480 grams. The motors constitute the largest single item.

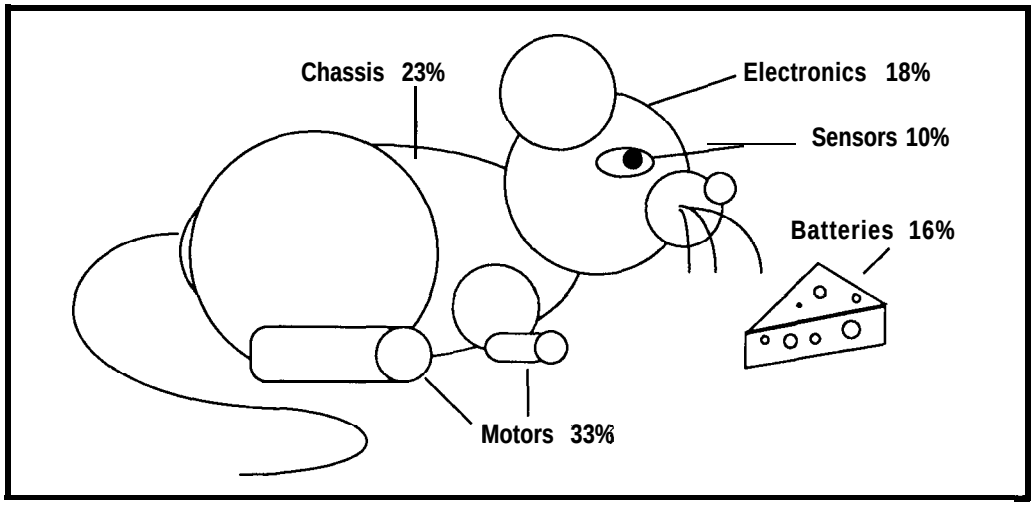


Figure 3—*MITEE Mouse III weighs a total of480 grams, with the largest single item being the motors.*

THE MOUSE'S MICROCONTROLLER

Many components are required to construct a micromouse, but at the heart of them all is a microprocessor to coordinate the systems and provide basicintelligence. After using the8051 microcontroller from Intel for MITEE Mouse I, we switched to theμPD78312 from NEC for MITEE Mouse II and III (see Figure 7a). *[Editor's Note: The complete MITEE Mouse III schema tic is in Figure 7 near the end of the article.]* This chip has most of the hardware needed for this design including 8K of on-board EPROM, two PWM generators, two 16-bit up/down counters, two 8-bit I/O ports, an 8-bit A/D converter, a serial interface, several timebase counters, interrupts with automatic register saving, and 16-bit multiply and divide instructions. The clock frequency is 12 MHz with the fastest instructions executing in 500 ns. An additional 32Kof external static RAM was added to the 256 bytes of RAM in the processor. Although a small part of RAM is used in the final version of the mouse program to store the maze configuration and other information, it is used primarily during development.

The 78312 has a serial interface than can be configured for use with shift registers or as a conventional UART. A 4-pin connector is provided on the mouse so that both modes may be used. When a monitor program is burned into theon-board EPROM, the UART is activated and a cable toahost computer is used to download programs into RAM and monitor their execution. When the final program is run, the cable is removed and the serial port is used to monitor the sensors. This provides a very convenient and compact system for developing programs. Though the programs are not generally more than 8K in length, a 32K RAM takes the same board space and approximately the same power as an 8K RAM, so it was used. The RAM chip is the only component on the processor bus, so only an address latch and no address decoding was used.

An additional use for the RAM chip is to store diagnosticinformationduring a run. In debugging the program, and even more in the optimization of the operation of the mouse,itisoftendesirable to see things through the eyes of the mouse. Tying a long umbilical cord to the mouse usually influences how it operatesand is undesirable. The large RAM can be used, however, to store the error signal on the motors or the signals from the sensors, perhaps even with some corrective action which the mouse took, and then read those signals out to a computer after the run is over.

To start the mouse at the beginning of a contest, a push button switch is included. An additional reset button is also provided to stop it if it crashes (hardware or software).

MOTOR MOUSE

To provide the highest performance motor drive, DC motors with servo position control were selected.

Figure 7b shows the complete schematic of the drive circuit. DC motors seem to have better power-to-weight and power-to-volume ratios than stepper motors. This more than compensates for the extra complexity in driving them. A PWM drive is used for each motor to maximize the efficiency of the system and minimize the heat sinking required. Because the processor has two PWM waveform generators built in, this type of drive also required very few parts.

The PWM signals are buffered with Teledyne TSC427 drivers and used to drive complementary International Rectifier P- and N-channel MOSFETs. The power supplies used on the mouse are nominally 12 volts, which allows the P- and N-channel FETs to have their gates tied together without exceeding the 20-volt maximum gate voltage specification. This allows them both to be directly connected to the output of the TSC427 whichprovides the level shiftingfrom the 5-volt logic supply to the 12-volt motor supply. This chip is also designed to drive the high capacitance of a power MOSFET gate. A potential problem with this type of configuration is shoot-through, where both the P- and N-channel devices are conducting at the same time, effectively shorting out the power supply. This problem is minimized, however, because the TSC427 is able to switch the power FETs through their linear regions in approximately 30 ns. The MOSFETs chosen have a low on-resistance of 0.28 and 0.20 ohms for the P- and N-channel devices, respectively. This allows efficient operation with the 1- to 2-amp motor currents anticipated. They are packaged in 4-pin half mini-DIP packages which makes them very compact. The entire drive electronics for both motors occupies the space of three 16-pin ICs.

Attached to the shaft of each motor is a Hewlett-Packard HEDS-9100 optical incremental encoder. The code wheel selected provides 512 pulses on each of two channels for each revolution of the motor. The two channels are 90 degrees out of phase allowing the direction of rotation to also be determined from the two signals. The 78312 has two 16-bit up/down counters with external up/down control that can be configured to interface directly with incremental encoders. Figure 4 shows a typical waveform from the encoder. Unfortunately the processor chip is not able to deal with the oscillations in the encoder signals that often result when the mouse stops and rocks back and forth just as the shaft encoder was about to make a transition to a new position. To overcome this problem, 4027 J-K flip-flops were added to debounce the encoder signals. The recently announced "A" version of the processor has solved this problem internally.

Accurate control for each wheel is necessary to steer the mouse, as well as to accelerate and decelerate it smoothly. A servo loop is therefore implemented for each motor. The processor calculates the desired position for each wheel on a millisecond-by-millisecond basis, measures the actual position from the up/down

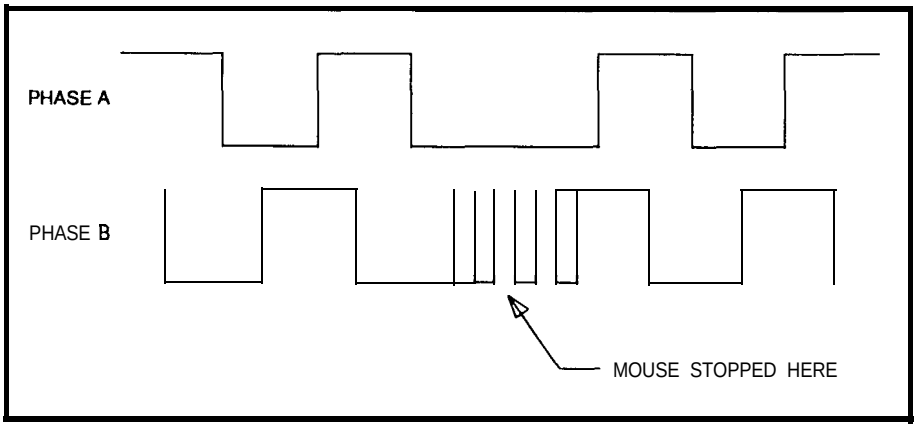


Figure 4—*Shaft encoders are connected to the* motors. *Oscillations sometimes occur if the mouse stops just as an encoder was about to make a transition, as shown in phase B.*

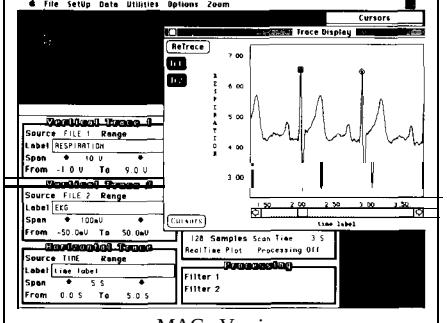
DATA ACQUISITION SOFTWARE

We'll never leave you without a trace

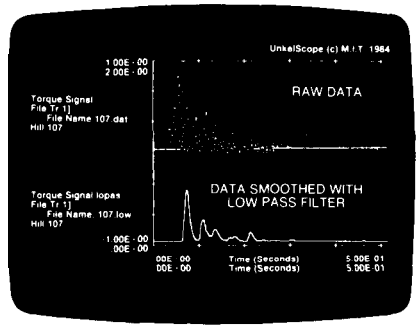
UnkelScope®

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

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



MAC Version



PC Version

UnkelScope JUNIOR $125
UnkelScope for MAC $149
UnkelScope Level 2+ $549
30-day money-back guarantee

(617) 861-0181
FAX (617) 861-1850

Unkel Software Inc.
62 Bridge Street, Lexington, MA 02173

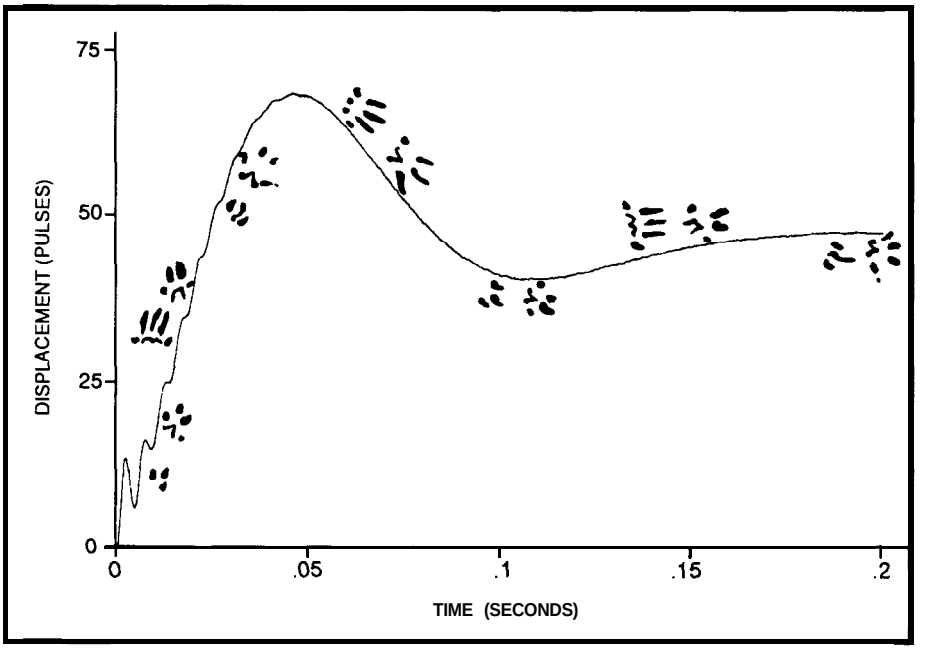


Figure 5—*The step response of the motor servo shows some high-frequency oscillations at the beginning due to the rubber tires.*

counter, and computes an error signal. This signal is then digitally filtered by the processor and the result applied as a voltage to the motor through the PWM amplifier. The 16-bit multiply instruction of the processor is very important for efficient implementation of the digital filter. To get good performance from the servo loop, compensation of the system is required, and this is provided by the digital filter. Figure 5 shows the step response of one of the motors. (The high-frequency oscillation at the leading edge of the waveform is due to additional dynamics introduced by the rubber tires used on the mouse.) Unfortunately, while the 16-bit multiply instruction is very helpful, the fact that it is not a signed multiply is very unfortunate. This is compounded by the fact that the processor does not have a 16-bit complement or two's complement instruction. The net result is that is takeslonger to figure out the sign of the result than to do the multiplication.

JUICE FOR THE MOUSE

Power for the mouse comes from four Duracell DL2/3A lithium batteries. These cells are designed for high-rate applications in auto focus, flash, and advance cameras, and can supply 1 amp continuously and 4 amps for short periods of time. While this is not much current compared to conventional NiCd batteries, it is very good for lithium batteries. Because lithium batteries have a much higher energy-to-volume and energy-to-weight ratio than conventional batteries, MITEE Mouse III was designed to use these cells. This meant that the weight of the mouse had to be kept low so that large currents would not be required to accelerateit.Thebatterieshavea 1300-mAH capacity which allows for over an hour of normal operation--considerably more than required since each mouse only gets 15 minutes to compete. Because the cells are expensive, however, the long life is a welcome feature.

Power for the drive motors comes directly from the battery. Two 1500-µF caps filter the supply and help to reduce the current ripple from the PWM drive. Because the internal resistanceof the batteries is not that low, it is important that the battery only see the average current and not the peak motor current. The peak motor current has a higher RMS value than the average motor current and this would result in greater losses.

The8-bit A/D converter built into the 78312 is used to continuously monitor the battery voltage and prevent mouse operation if the batteries are too low for reliable operation. (Mice attempting to run with low batteries often destroy themselves.) The exact batteryvoltageisalsousedin themotor servo loop to compensate for changes in feedback gain due to changes in battery voltage.

Power for the logic and processor is provided by a National 2951 voltage regulator connected to the same batteries used to power the motors. This low-drop-outregulatorusesverylittle quiescent current for long battery life, provides 5 volts without external components, is packaged in a compact mini-DIP package, and has a power fail output to interrupt the processor if desired.

NOT A BLIND MOUSE

A key system of the mouse is its sensors. The sensors are used for both

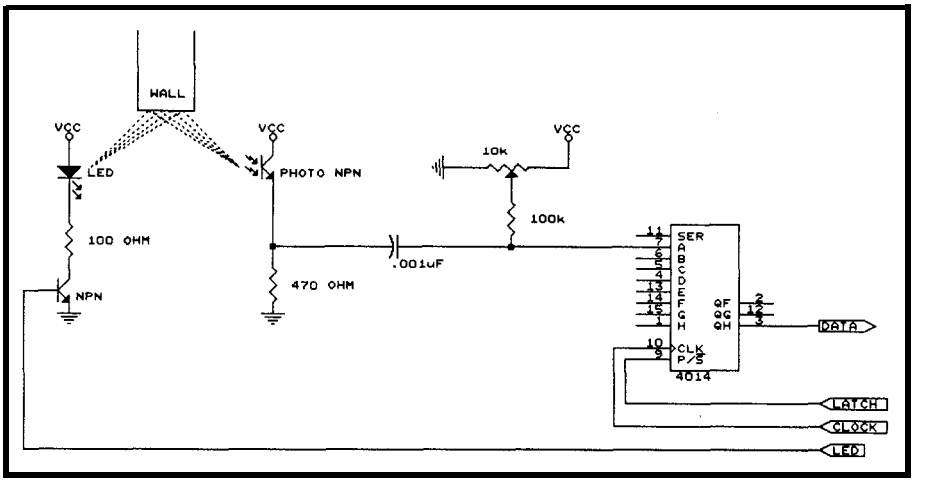


Figure 6—*The basic sensor circuit reflects infrared light off the tops of the maze walls to determine the locations of openings.*

local **navigation and** filling in the maze map in preparation for solving the maze. Local navigation involves the process of looking at the edges of the walls and correcting the heading or position of the mouse if it is not traveling down the middle of the maze. MITEE Mouse III uses four linear arrays of infrared sensors arranged to capture the position of the four posts at the comer of every square as the mouse travels through the maze.

Figure 6 shows a simplified schematic of the one sensor circuit. Each sensor consists of an 0P269 infrared emitter and OP509 infrared detector. The emitter and detector are placed sidebysideandarranged to lookdown at the floor of the maze from above the walls. Each device has a lens built into the plastic to limit the field of view of each sensor. In general, the light from theemitterisnot reflected by theblack floor but is reflected by the tops of the walls, which are quite close to the sensors and painted red. Ambient light can be very high if a mouse contest is televised, therefore the output of each detector is AC coupled to suppress the average signal. The 470-ohm load resistor is selected to prohibit saturation of the detector with the brightest anticipated ambient light. Figure 7c showsthecompletesensorarrayfound on MITEE Mouse III.

The detector signal is coupled into a 4021 shift register. The voltage on a 100k bias resistor is adjusted so that each input is several tenths of a volt below the threshold. The processor tumson theemitter, waits10µsforthe detector to respond, and then latches the detector signal into the 4021 and turns off the emitter. All the signals on all the detectors are simultaneously latched into the shift registersand then shifted into the processor serially. The 78312 has hardware support for handling data from shift registers, so that each byte can be brought into the processor under interrupt control without the time consuming task of **providing each** shift clock. Additional sensors can also be added if needed without the fear of running out of I/O lines on the processor. By only pulsing the emitter for 10 µs every 1 ms, high peak currents can be used for maximum sensitivity, and total power consumption is still low.

The synchronous detection system used here is also very good at

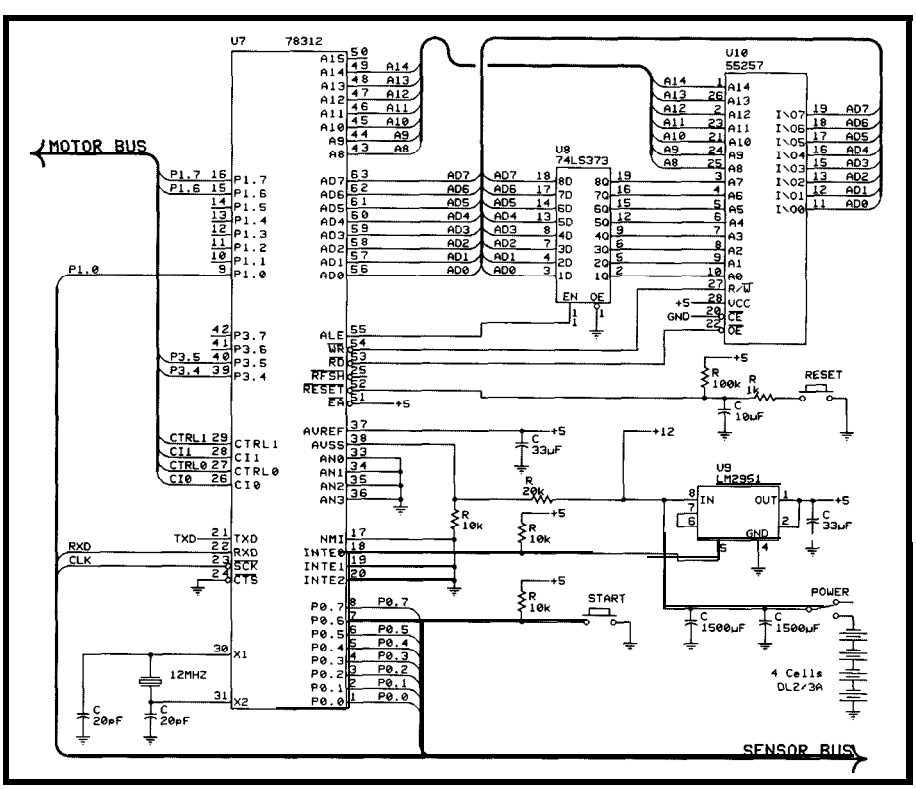


Figure 7a—*The brains of the micromouse consist of an NEC µPD783 12 microcontroller and 32K of external RAM.*

MINDSEYE TM

SYNERGIZER

Converts your PC to a Brainwave Entrainment machine



The MindsEye Synergizer is a powerful hardware/software combination that turns your IBM PCXT/AT/386 or clone into a laboratory grade audio/visual synchronizer. The Synergizer allows the user to program sessions of almost any length and complexity with each eye and ear programmed independently if desired, pulses can shift from one rate to another, while different sound frequencies are channeled left and right. Multiple time ramps and sound/light levels (over 32,000) may be included within a single programmed session. A stereo synthesizer makes available a variety of waveforms, filters and other sound parameters. The Synergizer provides more programmable capabilities than any other device available, at a remarkably low price. Requires DOS 3.0 or above; 512 K of RAM, and a hard drive are recommended.

MINDSEYE SYNERGIZER $330
Includes board, software, manual

MINDSEYE GOGGLES $65
Required for use with Synergizer

EXTERNAL CONTROL UNIT $95
Software-assignable controls

SONY MDRP1 HEADPHONES $35
Standard MiniStereo plug

SYNETIC SYSTEMS TM
MIND TECH FOR A NEW AGE

ORDER PHONE
800-388-6345
Credit Cards Accepted

Information (206) 789-6345
PO Box 95530
Seattle, Washington 98145

© 1990

Reader Service #162

suppressing noise. If someone should accidentally take a flash picture of the mouse while it is running (expressly forbidden at a contest), it is unlikely that the flash will be synchronized with the sampling of the sensor by the processor. In practice MITEE Mouse III seems to be very immune to this type of problem.

PACKAGING IMPORTANCE

On MITEE Mouse I we used wire-wrap sockets for the processor circuitry. Unfortunately sockets weigh twice as much as chips, so that this type of construction weighs three times what it needs to. MITEE Mouse III uses one double-sided 3" x 5" x 1/32" printed circuit board to mount the processor, motor drive electronics, and sensor circuitry with the exception of the emitters and detectors. The 78312 is an EPROM chip which will always be mounted in a socket, so the crystal, the address latch, and several other components were mounted under the chip in space that would normally be wasted. Extensive use of SIP resistor packs in the sensor circuitry also facilitated the packaging. because the board is so small, 1/32" thick material was used instead of the conventional 1/16". The aluminum frame of the

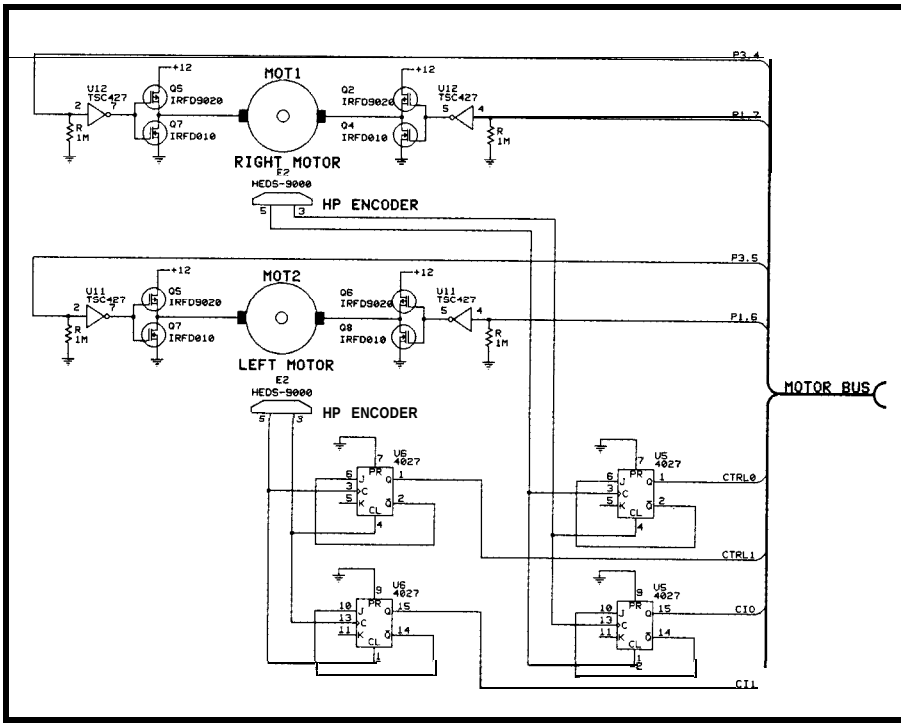


Figure 10- The complete motor control circuit shows the motor drive transistors and the shaft encoders.

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

R&D ELECTRONICS
1224 Prospect Avenue
Playhouse Square
Cleveland, OH 44115

Telephone: **(216) 621-1121** • Fax: **(216) 621-6628**

Reader Service #155

68000

68000 Debug Monitor
- Suitable for embedded installation
- Off-the-shelf hardware
- Locate anywhere in memory
- Full-feature memory and register commands
- Breakpoints and tracing
- Interactive assembly and disassembly

68000 Soft Tools
- Absolute cross assembler
- Linking cross assembler
- *C cross* compiler
- Design and reference material

Custom Development
- Hardware and software engineering design and prototyping available.

MICRO RESOURCES
Making Technology Work
717-524-7390 or
30 SOUTH EIGHTH STREET. LEWISBURG, PA 17837
717-523-0777

Reader Service #1

mouse extends beyond the PC board in the front and back so that even in a crash, the thinner board has not been a problem. Overall, the electronics (exclusive of the sensors themselves) contribute only 18% of the weight of the mouse.

The sensor emitters and detectors are mounted on a 1/64" printed circuit board which is then glued to a balsa wood frame. As mentioned previously, a great deal of effort went into minimizing the weight of the sensor assembly to keep the inertia of the mouse low. Fortunately the sensors are mounted above the walls of the maze so that when the mouse crashes, the sensors are never involved in the impact. As a result, the balsa wood frame has adequate strength for the application. A 29-pin connector is used to bring the sensor signals to the main circuit board. This allows the sensors to be removed easily for repair or modification.

. ..AND THERE'S MORE

MITEE Mouse III was first raced in February of 1988 and came in second. At that time it did not have the capability to do diagonals. This feature was first demonstrated in London in July where it came in second again. Since then it has raced and placed as follows:

July 1988	London	Second
October 1988	Montreal	First
November 1988	Anaheim	First
November 1988	Tokyo	Third
March 1989	Baltimore	First
July 1989	London	Second
October 1989	Singapore	Third
November 1989	Tokyo	**Second**
March 1990	Los Angeles	First

MITEE Mouse III has the honor of being the world's first mouse to be able to directly navigate diagonals in the maze.

The next article will focus on the software side of MITEE Mouse III. We'll explore the algorithms used to drive the motors, convert the sensor signals into useful information for navigation and maze mapping, and determine the optimal path given a maze configuration.+

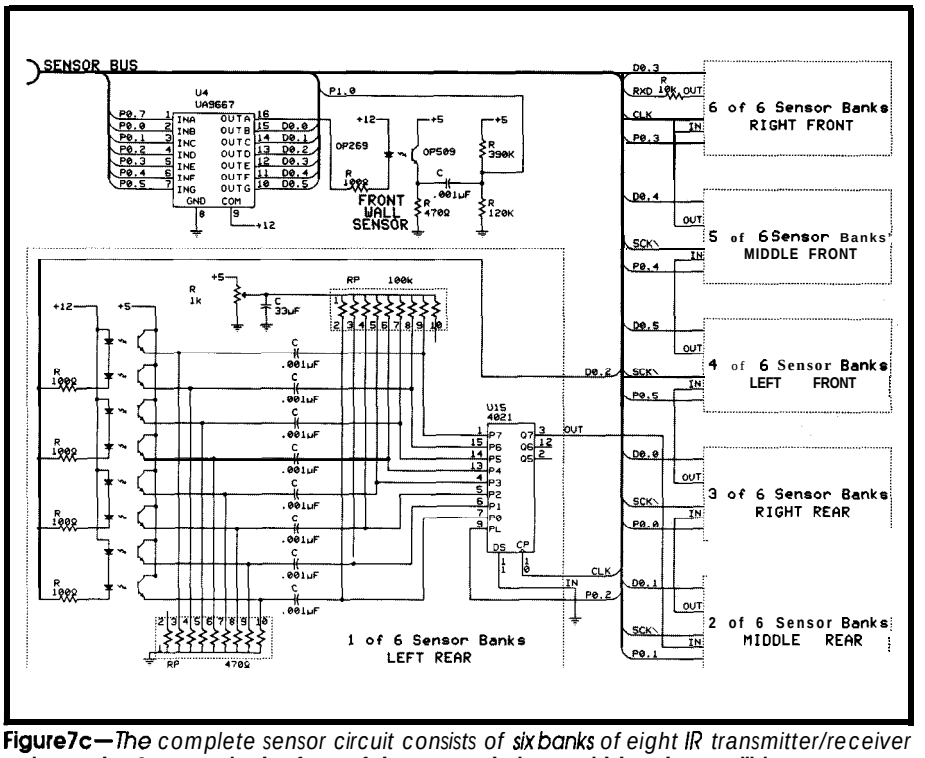


Figure 7c—*The complete sensor circuit consists of six banks of eight IR transmitter/receiver* **pairs each. A sensor in the front of the mouse helps avoid head-on collisions.**

David *Otten* is a principle *research engineer* in the *Laborato* y for *Electromagnetic* and Electronic Systems at *the Massachusetts* Institute *of Technology. He holds a B.S. and M.S. degree from MIT.*

IRS

2 10 Very Useful
2 11 Moderately Useful
2 12 Not Useful

The DA/M™

The Lowest Cost Data Acquisition System

Our DA/M can solve more of your data acquisition problems at a lower price than any other product on the market. DA/M's are used for:

- Military meteorological stations.
- Building management.
- Automated hydroponic farming.
- Industrial process control.
- Your application

In fact, DA/M's can be used whenever you can't afford to use any one else's product.

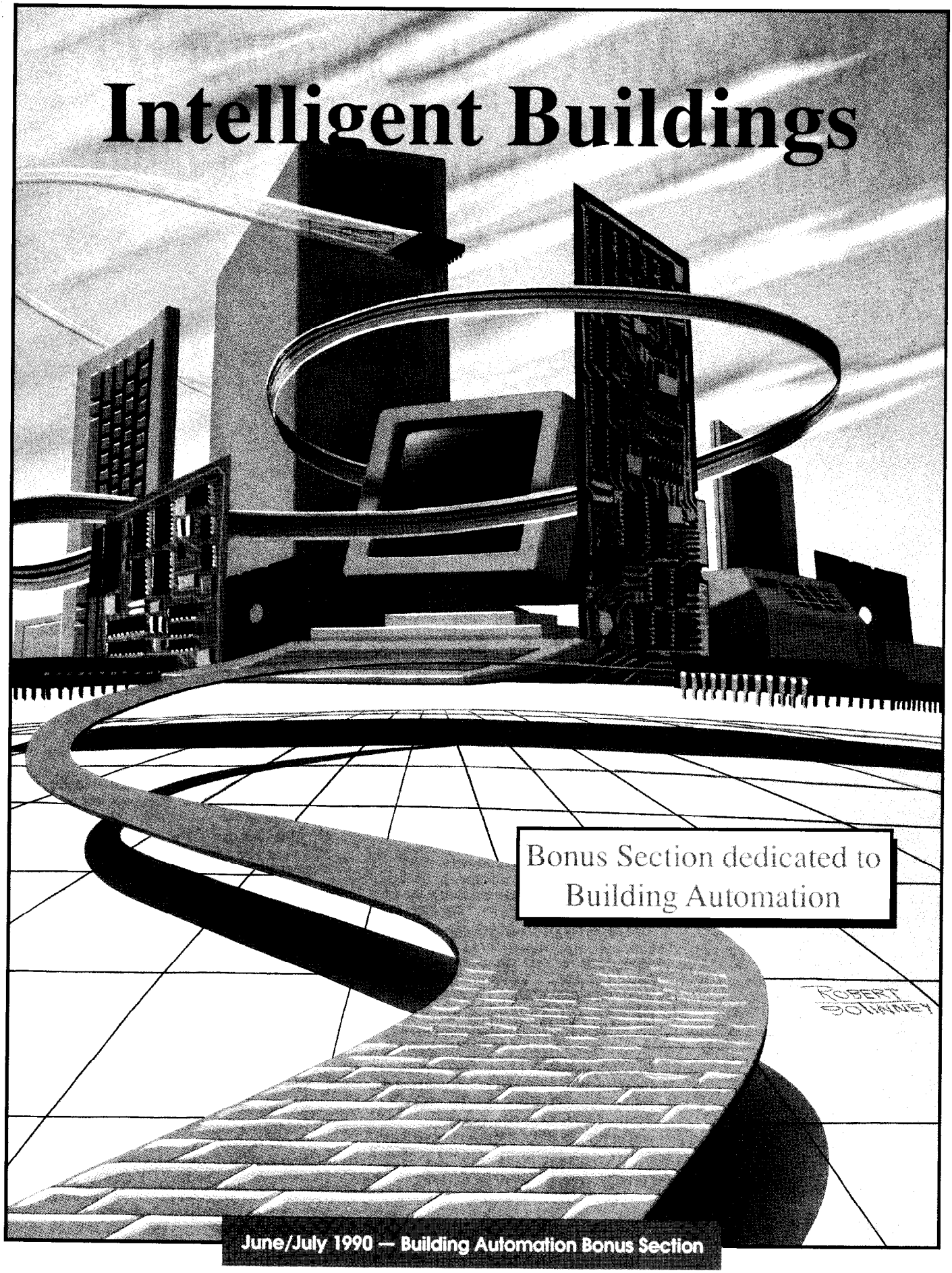
Made in North America, DA/Ms are available NOW!

DA/M
DATA ACQUISITION AND MANAGEMENT CORPORATION LTD.
#140, 17303 · 102 Avenue
Edmonton, Alberta Canada T5S 1J8

Phone 1-403-486-3534
Fax 1-403-486-3535

Reader Service #120

Intelligent Buildings



Bonus Section dedicated to Building Automation

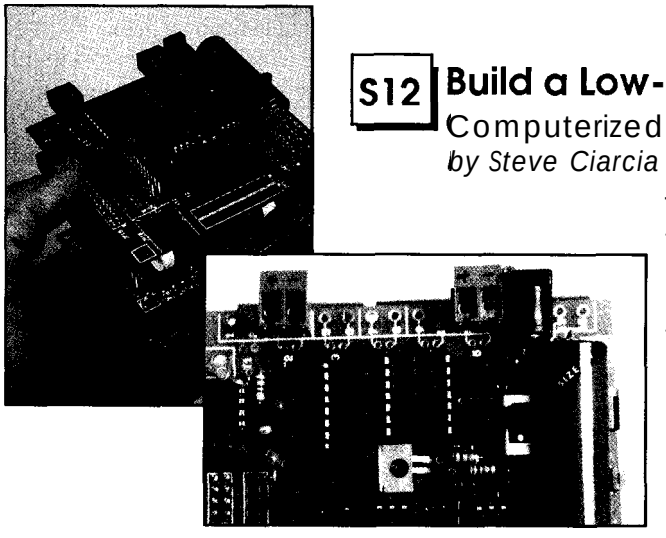
June/July 1990 — Building Automation Bonus Section

CIRCUIT CELLAR INK®
THE COMPUTER APPLICATIONS JOURNAL

Building Automation Special

S2 **CEBus Update**
How is the Health of EIA's Baby?
by Ken Davidson

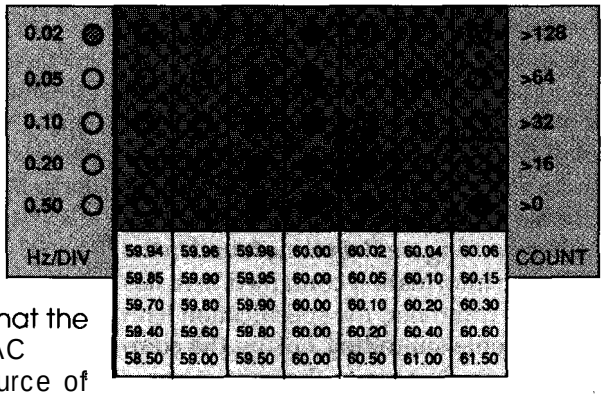
The preliminary standard is out, and now the *real* fun begins! Managing Editor Ken Davidson has gone inside the CEBus committee to bring back the latest news on the state of the home automation industry's most important standard.



S12 **Build a Low-Power Data Logger**
Computerized Data Collector Runs For Years on a Battery
by Steve Ciarcia

The most crucial decision in the automation process is the first: to automate or not to automate? Making an intelligent decision requires real data, and Steve Ciarcia shows you how to get it in this hands-on construction project.

S24 **Build a Power Frequency Monitor**
Counting Cycles Until it Hertz
by Ed Nisley



We take a lot for granted: the sun will shine, taxes will go up, and AC will come in at 60 Hz. Ed Nisley has checked it out, and shows that the last assumption may not always be true. His AC frequency monitor can help you trace the source of those mysterious motor problems and timing fluctuations.

FEATURE ARTICLE

Ken Davidson

CEBus Update

How is the health of EIA's baby?

There has been a revolution going on in the home automation arena these days. Nothing as bloody as the American Revolution, and nothing that will change society quite as much as the industrial revolution, but it's a revolution just the same. That revolution is CEBus.

Soon to disappear are the days of complex home control systems made up of components that must come from just one manufacturer if they are to compatible with one another. You won't need an engineering degree to install a simple system. And home owners will be able to get their feet wet in the home control pond by starting small, then expand their system without throwing away what they already have.

Revolutions don't come without pain, though, and they don't happen overnight. The Electronic Industries Association (EIA) pulled together a committee to develop a unified home automation standard close to six years ago, and after great pain (on some of the committee members' parts, anyway), much of their effort has come to fruition.

In the August/September 1989 issue of CIRCUIT CELLAR INK (#10), I described in detail the state of the CEBus specification according to the information available outside the committee at that time. Much of the information dated back to the summer of '88, and some had been put in place temporarily so the committee could assemble a working CEBus booth to display at the 1989 Winter Consumer Electronics Show, the National Association of Home Builders Show, and several shows since then.

Portions of the actual specification have since been released by EIA for comment, and an update to my original article is warranted. While I have no intentions of trying to duplicate the spec in its entirety in these pages, I do want to give enough detail to get the flavor of CEBus across to the engineer who may be toying with the idea of including a CEBus interface in his next product.

I will also include a warning similar to one found in the original article: While the information contained here has been released by EIA for comment, the specification isn't in its final form. Comments submitted during the comment period may influence changes to the spec at some point before it is adopted as a final standard. If you are considering using CEBus in a product, do not try to use this article as a source on which to base your design. Contact EIA directly to get a copy of the spec and to get more details concerning the anticipated timeframe in which the standards-making process is working.

THE ISO/OSI 7-LAYER MODEL

The CEBus standard (also know as the EIA Home Automation Standard or, as the public at large will likely come to know it, Synq) is based on the ISO/OSI seven-layer network model. Within that model, a network is broken into seven functional pieces, each having responsibility for one part of the network communication. At the highest level is the user interface, while at the lowest level is the actual physical medium which is carrying the communications (see Figure 1).

Each layer communicates with the layer above it and the layer below it through a set of well-defined "service primitives." The lower layers provide services to the upper layers, while the upper layers "subscribe" to the services of the lower layers.

CEBus doesn't use all the layers defined by the OSI model, and subdivides some of those it does use. At the highest layer is the Application Layer, which is where CAL, or Common Application Language, is found. CAL provides a language through which manufacturers may communicate with other devices on the network.

The Presentation, Session, and Transport Layers aren't used at all. Their intended functions either don't apply to the spirit of CEBus, or are incorporated in the other layers.

At the next lower level is the Network Layer which is responsible primarily for router control. Routers are used to connect portions of the network together which normally have no physical connection between them, such as between different physical media (e.g., to route messages from the power line to twisted pair). The Network layer also handles the sequencing of segmented packets.

The Data Link Layer is actually broken into two sublayers: the Logical Link Control (LLC) sublayer and the Medium Access Control (MAC) sublayer. The LLC receives a message from the Network Layer, adds a header to the message, and sends it to the MAC. The MAC is responsible for putting the message out onto the network. While the LLC is the same regardless of the physical medium used, the MAC may be different since different media often require different access methods.

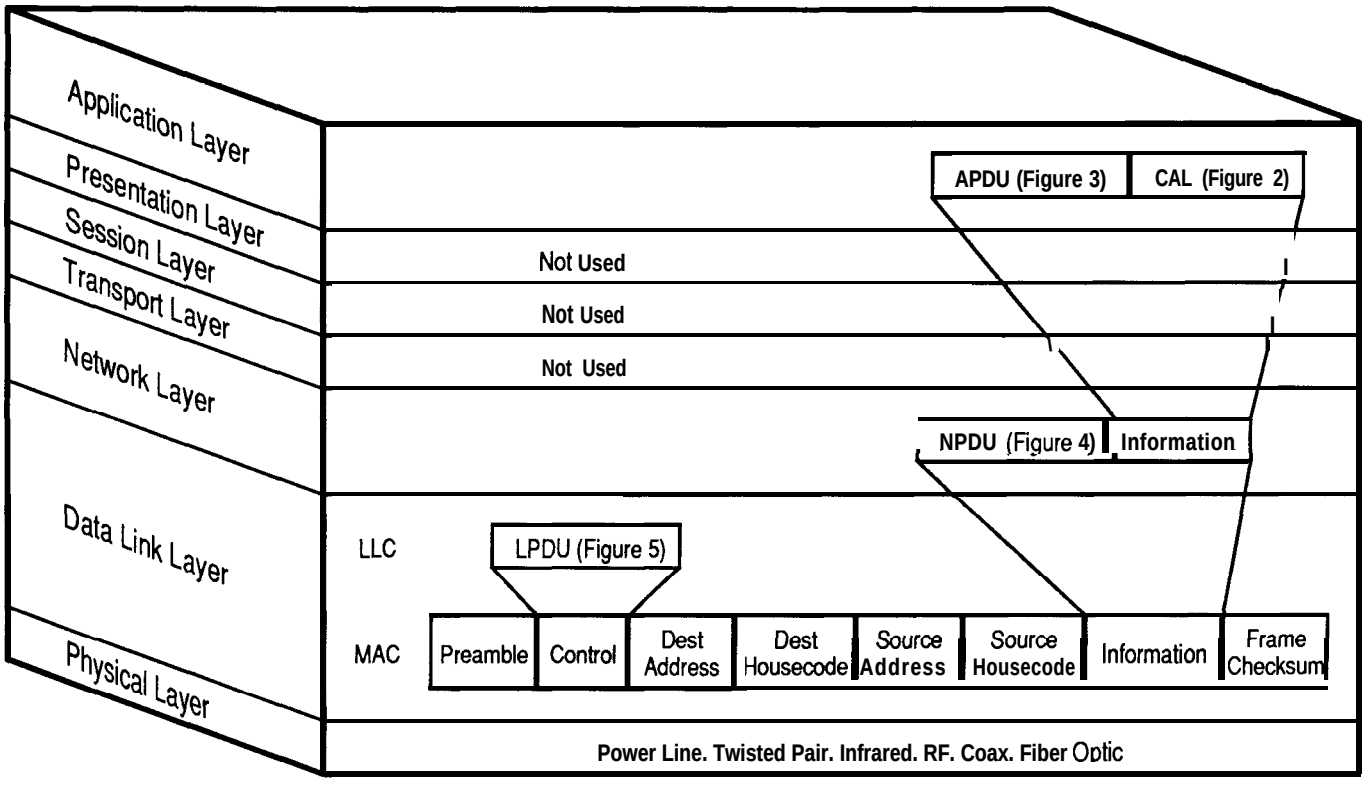


Figure 1 —*CEBus is based on the ISO/OSI seven-layer network model. Within thatmodel, a network is broken into seven functionalpieces, each having responsibility for one part of the network communication.*

At the lowest level is the Physical Layer, which takes care of actually transmitting the message over a physical medium. The CEBus specification defines six different physical media, including power line (PL), twisted pair (TP), coax (CX), infrared (IR), radio (RF), and fiber optic (FO). TP, CX, and FO are often collectively referred to as wired (WIBus), and IR is also called SRBus (single room). While fiber optic has been included with an eye to the future, virtually no work has been done on fiber optic, so I won't mention it again.

Overseeing the whole shootin' match is Layer System Management, which takes care of initializing each layer and maintains each layer's peer-to-peer protocol.

The point of using a layered model is that it's is easy to make the upper network layersidentical for all nodes, with different lower layers which depend on the physical medium being used.

Commands or data to be sent from one node to another originate in the Application Layer. That data works its way down through the layers until it is transmitted onto the physical medium by the lowest layer. Once received, the data makes its way back up through the layers and ends up at the Application Layer on the destination end. Let's follow that data packet from the Application Layer down through the Physical Layer.

THE APPLICATION LAYER AND CAL

At the highest layer of CEBus is CAL, or Common Application Language. CAL is a complete control language which CEBus devices use to communicate with one another. Its primary function is twofold: to allocate resources and to perform control.

Resource allocation deals with requesting, using, and releasing resources within CEBus. For example, some of the media include not just a control channel, but several data channels. CAL commands have been defined for allocating individual data channels to requesting devices.

The main use of CAL initially will be for control. The language has numerous commands defined for turning devices on and off, dimming up and down, opening and closing, plus more complicated actions such as setting VCR presets or responding to telephone commands.

At its core, CAL is actually table driven. There are tables of constants which have been defined to represent device categories, devices themselves, commands, actions, and responses. As new devices are developed by manufacturers, the tables will be expanded (under EIA's control) to include those devices and any new functions that may be associated with them. Suppose, for example, someone develops a robot vacuum cleaner that needs to be CEBus controlled. There currently aren't any robot vacuum cleaner commands in the CAL tables, so the manufacturer would approach EIA with a list of required commands to be added. Any devices developed after the commands are added could incorporate robot vacuum control in their repertoire.

No discussion involving modern languages is complete without somehow bringing object-oriented programming (OOP) into the picture. In response to a last-minute

a)

00	Abstract Utility	64
30	Audio Process	65
31	Audio Source	66
32	Audio Record	68
38	Video Monitor	
39	Video Source	69
3A	Video Record	
40	Tuning System	
48	Time Service Element	
60	Appliance Control System	

b)

80	FALSE	99
81	TRUE	9A
84	TEST	9B
85	COMPARE	9C
86	ADD	
87	ADD-immediate	
88	SUBTRACT	
89	SUBTRACT-immediate	
8A	LOAD	
8B	LOAD_immediate_num	
8C	LOAD_immediate_strng	
8D	STORE	
8E	SWAP	B1
8F	NO OPERATION	
90	AND	
91	OR	B9
92	XOR	
93	NOT	
94	BRANCH	
95	BRANCH_conditional	
96	JUMP	
97	CALL	
98	RETURN	

Figure *2-a) The defined tly b)nge of today's home electronics and appliances.*

(Backus-Naur contains complete BNF

you end up with a string of bytes context, an object, a method, and an optional list of arguments.

"Contexts" define what the device is that is being referenced. As Figure 2a shows, the present spec covers the gamut of devices commonly found in the home today. Unused values are reserved for future definitions.

"Objects" typically define a switch, button, or knob on the device. "Primary mode switch" might be the main power switch, while in the case of a preamp, "source switch" might select between the system tuner and the CD player.

"Methods" fall into one of six categories: boolean, arithmetic, data transfer, logical, control transfer, and other. These are what define the action to be taken. Figure 2b contains the methods defined at present. Like the rest of CAL, there is plenty of room for expansion.

The CAL command ends with an optional list of arguments that further define the action to be taken or supply additional information to the device. For example, a command to set a clock will contain the present time within the argument field.

Once the CAL command has been assembled, the message transfer section of the Application Layer adds a header to the front of the data to create an APDU (Application Layer Protocol Data Unit). There are six APDU modes, with header sizes ranging from one byte up to a variable number of bytes. Rather than trying to cover all the possibilities, I'll show just the simplest APDU modes.

The "B1F" mode uses basic (unprivileged) service class with a one-byte, fixed header and is shown in Figure 3a. Four valid types in this mode include explicit invoke (requires a response from the receiver), implicit invoke (doesn't need a response), result, or error. The header also includes room for an invoke ID, which may be used to tie a response to a particular command (since multiple commands may be outstanding).

The "P1F" mode, shown, in Figure 3b, is identical to B1F but uses privileged service class. Basic service is used for node-to-node communication between Application Layers, while privileged is used to communicate between Layer System Management entities.

THE NETWORK LAYER

The APDU is passed from the Application Layer to the Network Layer, where another header is added to the front

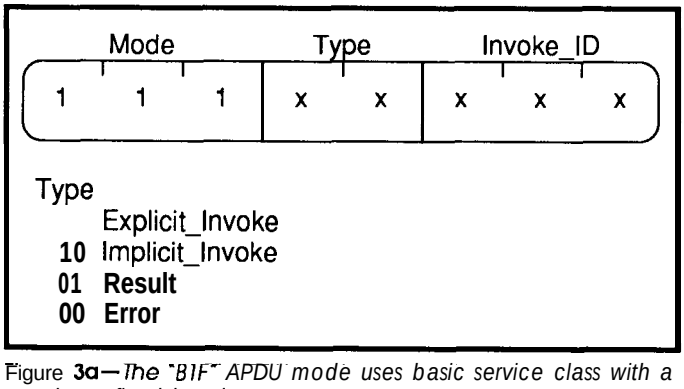


Figure **3a**—*The "B1F" APDU mode uses basic service class with a one-byte, fixed header.*

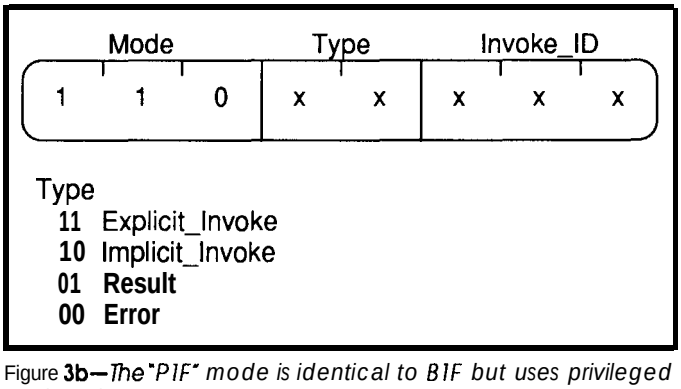


Figure **3b**—*The "P1F" mode is identical to B1F but uses privileged service class.*

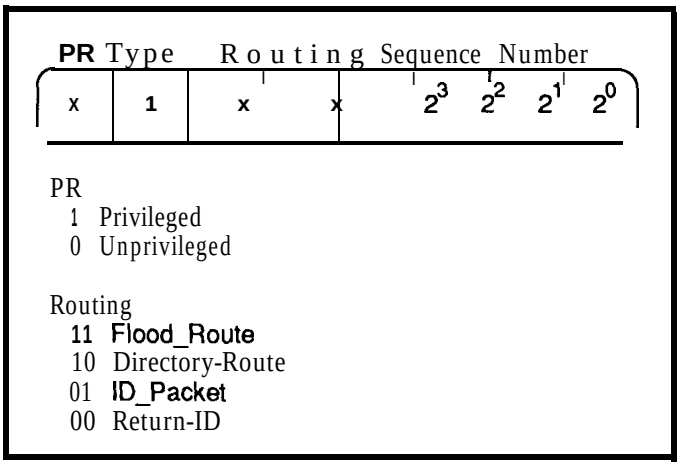


Figure 4-Normal-formatted *NPDUs* are used *for* unsegmented data with either directory or flood routing.

of the packet, resulting in an NPDU (Network Layer Protocol Data Unit). Two NPDU formats are defined: normal and extended.

Normal-formatted NPDUs are used for unsegmented data with either directory or flood routing. In directory routing, a packet which is to be routed to a node on a different medium is retransmitted only on the medium to which the destination node is connected. Each router contains a directory of nodes and where they are located, so such intelligent processing is possible. In flood routing, the packet is retransmitted on all media in use.

As with the APDU, privileged and unprivileged NPDUs are available. Privileged NPDUs are originated by a call from the Application Layer, while unprivileged NPDUs are sent by the Layer System Management.

Extended NPDUs are primarily used when a packet must be broken into several pieces, and handle the resulting need for segmentation and flow control. They also support special messages utilized by routers to determine network topology.

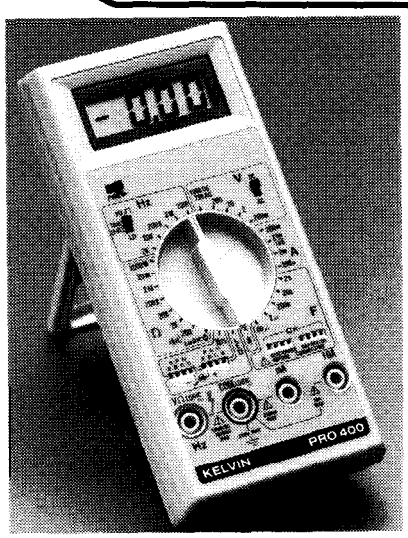
The format of a normal NPDU is shown in Figure 4. Extended NPDUs can get complicated and won't be covered here.

THE DATA LINK LAYER

The Data Link Layer is broken into two sublayers: the LLC and the MAC. NPDUs coming down from the Network Layer pass into the LLC, where, again, a header is added to the front to form an LPDU (Logical Link Control Sublayer Protocol Data Unit).

The LPDU has a fixed format, as shown in Figure 5, and may be one of five types: local acknowledged, local unacknowledged, nonlocal acknowledged, acknowledge, and failure. The first three types are outgoing packets containing data; the last two are responses to received data.

In **unacknowledged** service, a packet is sent out blindly in the hopes that it makes it to its destination. The receiver

DON'T PAY RETAIL - GET IT WHOLESALE!
POSSIBLY, THE BEST METER IN THE WORLD.
DEFINITELY, THE BEST VALUE!
YOUR SATISFACTION IS GUARANTEED
with 20MHz FREQUENCY COUNTER
EXCELLENT FOR
COMPUTER & VCR REPAIR
KELVIN
PRO 400
☑ LOGIC TEST
☑ DIODE CHECK
☑ CONTINUITY TESTER
Ed FREQUENCY COUNTER
☑ 5 FREQUENCY RANGES
☑ TRANSISTOR hFE TEST
☑ 5 CAPACITANCE RANGES
Ed DISPLAY - 3 1/2 DIGIT LCD
Ed LED TEST VERIFY GOOD/BAD
☑ AC AND DC VOLTAGE RANGES
Ed ACANDDCCURRENTRANGES
☑ TROUBLE SHOOTING TO 20MHZ


LIMITED TIME!
INTRODUCTORY OFFER
$69.95
Stock No. 990092
30 DAY MONEY BACK
GUARANTEE!
PROTECTIVE CARRYING
CASE for PRO 400
Stock No. 990094 $9.95 ea
KELVIN Electronics
7 Fairchild Ave. Plainview, NY 11603
Call: 1 (800) 645-9212
1 (516) 349-7620
FAX: 1 (516) 349-7830
GET IT WHOLESALE!
RESISTORS, IC's, LED, PRINTED CIRCUIT
MATERIAL, SOLDERING SUPPLIES
CIRCLE RESPONSE CARD FOR
FREE CATALOG

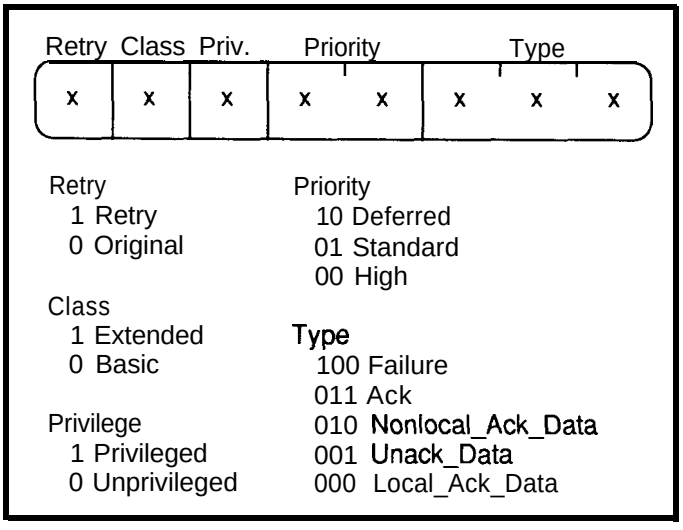


Figure 5—*The LPDU has a fixed format and may be one of five types: local acknowledged, local unacknowledged, nonlocal* acknowledged, acknowledge, *and failure.*

doesn't send any verification of receipt. In acknowledged service, the receiving node must verify to the sender within a fixed amount of time that the packet was received error free. If a sending node doesn't receive an acknowledge within that fixed amount of time, the packet is resent. If the packet isn't acknowledged after the second try, an error is sent up to the Network Layer. A failure response is sent by a receiving node when the packet arrived error free, but the node can't accept the packet for some reason.

Packets may have one of three priorities: high, standard, or deferred. The priority is used by the MAC to determine the channel access delay, or the delay between the channel becoming free and the beginning of transmission. Higher priority packets have first crack at the channel and therefore have a better chance of getting through.

As with the other PDUs, the LPDU may be either privileged or unprivileged. Unprivileged LPDUs originate in the Network Layer; privileged LPDUs originate in the Layer System Management.

At this time, all LPDUs use a basic service class. Extended service class is reserved for future use.

Finally, a bit is used to denote whether the packet is the original, or the second copy sent if an acknowledge isn't received. If the original packet is acknowledged by the destination node, but that acknowledgeislost, the sending node will think the original packet was lost and will send it again. Thedestinationendsupwithasecondcopyofthe same packet. The original/retry bit allows the destination node to differentiate between the two so it can throw the second copy into the bit bucket.

The LPDU is passed down from the LLC to the MAC sublayer, where the MAC adds some more information onto the packet to create the MAC frame. As Figure 1 shows, the final packet is made up of preamble, control, destination node number, destination house code, source node number, source house code, information, and checksum fields. The control and information fields come from the LPDU; the MAC adds the rest.

Once the MAC frame is complete, it's time to send the packet. The basis for channel access in CEBus is CSMA/CD, or Carrier Sense Multiple Access with Collision Detection. Nodes first listen to make sure the channel is quiet. If so, they begin their own transmission, listening as they transmit. If another node begins transmitting at the same time, a collision occurs. Typically, one of the two colliding nodes won't hear the collision since it is transmitting, and it continues sending to the end of the packet. In order to ensure that the information in the packet isn't corrupted by the collision, a preamble is added to each packet in the form of a pseudorandom 8-bit number. Since the **preamble** is thrown away at the destination (and not included in the checksum), if any part of it is corrupted by a collision, the packet will still arrive without errors. Use of a pseudorandom number maximizes the chances of detecting collisions during the preamble (two nodes won't be transmitting the same bit stream, which would make collision detection impossible). The node that backed off from the collision waits for the other node to finish, then tries again for control of the channel.

The MAC has a number of schemes it uses to ensure fair access to the medium for all nodes on the network. A channel access wait time is assigned to the node which depends on the pending packet's priority, whether the node's last attempt to send a packet was successful, and a randomization scheme that **adds a** random amount of time (Figure 6).

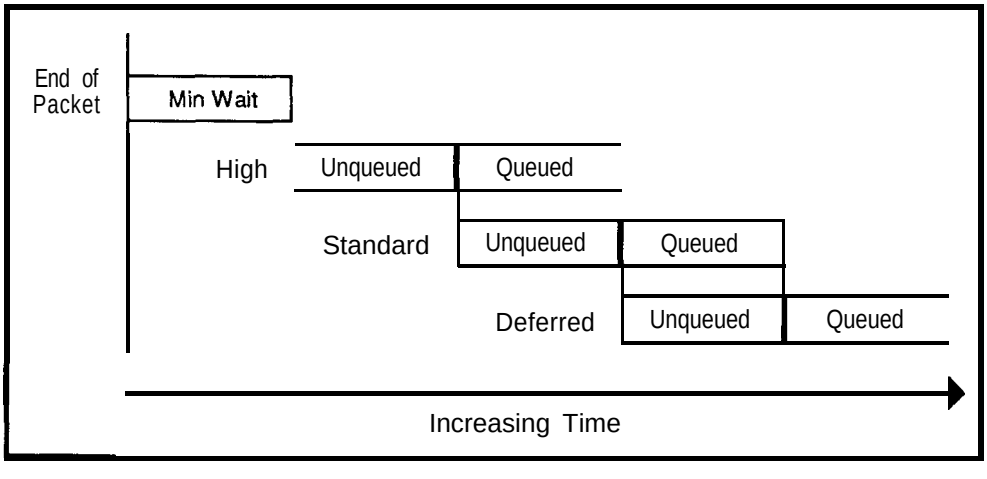


Figure 6—*The MAC uses a number* of *schemes to ensure fair access to the medium, including priority, last successful transmission, and a randomization factor.*

High-priority packets have a shorter channel access wait time than do standard- or deferred-priority packets.

When a node successfully transmits a packet, the node is placed in a queued state, which adds a small amount to the channel access wait time for the next packet. If the node is unsuccessful in transmitting a packet, the node goes to an unqueued state and extra wait time isn't added.

In addition to the schemes based on priority and success rate, a random amount of time is tacked on to prevent nodes which have fallen into the same priority/queue state from transmitting at the same time.

THE PHYSICAL LAYER

The lowest network layer is the Physical Layer and is responsible for the actual generation of signals that are transmitted onto the physical medium. In theory, this is the only part of the network model which is different between nodes depending on the medium being used. In reality, there may be a few differences in the MAC as well, but all the layers above the MAC are identical. The physical layer is broken down into two sublayers: the Physical Layer Symbol Encoding (PLSE) sublayer and the actual hardware interface.

There are four physical layer symbols defined for all the media: 1, 0, EOF (end of field), and EOP (end of packet). Each is defined in terms of a unit symbol time (UST) whose length depends on the medium being used. A "1" is one UST long, a "0" is two USTs, an EOF is three USTs, and an EOP is four USTs.

In order to shorten packet length as much as possible, leading-zero suppression is used on all fields except the preamble. As a result, it isn't possible to simply count bits to determine where one field ends and another begins, so an EOF symbol is sent to separate fields. An EOP, obviously, is sent at the end of the packet.

In addition to the four symbolsdescribed above, a fifth symbol called the "null" symbol is defined for the power line. More about the null symbol in a bit.

As mentioned before, there are several media addressed by CEBus, including power line, twisted pair, coax, IR, and RF. Only the power line portion has been released by the CEBus committee, however work is proceeding on the other media. Expectations are that one or more of the other media will be ready to be released for comment before the end of 1990.

POWER LINE

Perhaps the most useful medium of the six defined CEBus media for retrofit installations is power line. Virtually all buildings in North America are wired for AC power, making it the medium of choice for low-cost wired applications. The biggest drawback of the power line, though, is that it's a harsh and noisy environment, making error-free high-speed communication using conventional

1990s—
THE HOME AUTOMATION ERA

WHY WAIT?

HAA
HOME AUTOMATION ASSOCIATION
"Trade Organization of the Home Control Industry"

JOIN THE HOME AUTOMATION ASSOCIATION

Pioneer thedevelopment of this new growth industry along with industry leaders. Why watch the market when you can shape it?

Participate in trade conventions, exhibits, workshops, and seminars. The Association will provide information on trends, forecasts, and market activities to support more intelligent marketing decisions.

The Association also plans to promote home automation in the widest sense, monitoring the current state of the art as well as the social, economic, legal, and other effects of home automation on every level of society.

For futher informationonmembership, please contact Roy Mason at:

Home Automation Association
Post Office Box 3731
Georgetown Station
Washington, DC 20007
(202) 333-8579

Reader Service #182

QUALITY PARTS • DISCOUNT PRICES • FAST SHIPPING
ALL ELECTRONICS CORP.
P.O. Box 567 • Van Nuys, Ca • 91408

0 - 6 HOUR AUTO SHUT-OFF TIMER
M.H.Rhodes, inc.
Mark-Time# 90007 Wall-box timer fits standard 3" deep wall box. Rated 20 amps @ 125 Vac. Turn knob to desired time. Includes hardware. beige wall plate, and knob. UL and CSA listed. CAT# TMC-6
35.75 each . 10 for $50.00

INSTRUMENT ENCLOSURES
Molded ABS instrument enclosures. Matching front and rear panels. Integrated PC board standoffs and two sets of vertical mounting slots for front and rear sub panel PC boards. All enclosures are 6" wide X 6 1/4" deep. Available in black, ivory, blue, and beige. Specify color.
FRONT & REAR PANEL HEIGHT
2 1/4" CAT# MB-A $7.50 each 10 for $65.00
2 5/8" CAT# MB-B $7.75 each 10 for $67.50
3" CAT# MB-C $8.00 each 10 for $70.00

THUMBWHEEL SWITCH
1 pole 10 position decimal encoded switches which interlock to make up desired number of digits. Terminates to 11 pc pins (1 common and 10 poles). Each section is .31" wide X 1.20" high X .78" deep. CAT# SWTH-9 $1.25 each 10 for $10.00 • End plates can be added to form a .94" high bezel. END PLATES - CAT# SW-9EC- $1.00 per set

10 AMP SOLID STATERELAYS
ELECTROL# S2181
CONTROL: Rated 5.5 to 10 Vdc (operates on 3-32 Vdc). LOAD: 10 amp @ 240 Vac
2 114' X 1 3/4" X 7/8" CAT# SSRLY-10B
$9.50 each QUANTITY DISCOUNT
10 for $85.00 • 25 for $175.00
50 for $300.00 • 100 for $500.00

PHOTOFLASH CAP.
Rubicon CE 210 MFD 330 V 0.79" dia X 1.1' high. New, prepped with 1.4' black and red wire leads soldered to the terminals.
CAT# PPC-210 $2.50 each
10 for $22.50 • 100 for $200.00

OPTO SENSOR
U shaped package with mounting ears.1/8" opening. 3/4" mounting ears. CAT# OSU-6
50¢ each • 10 for $4.50 • 1 00 for .s40.w

ORDER TOLL FREE 1-800-826-5432
FAX (818) 781-2653 Terms: Phone orders must be charged to Visa, MasterCard, or Discover. Minimum order $10.00 • Add $3.50 per order for shipping/ handling • CA residents add sales tax • Quantities limited • Mail orders may be paid by check • No C.O.D.
CALL TOLL FREE FOR A FREE 60 PAGE CATALOG !

Reader Service #175

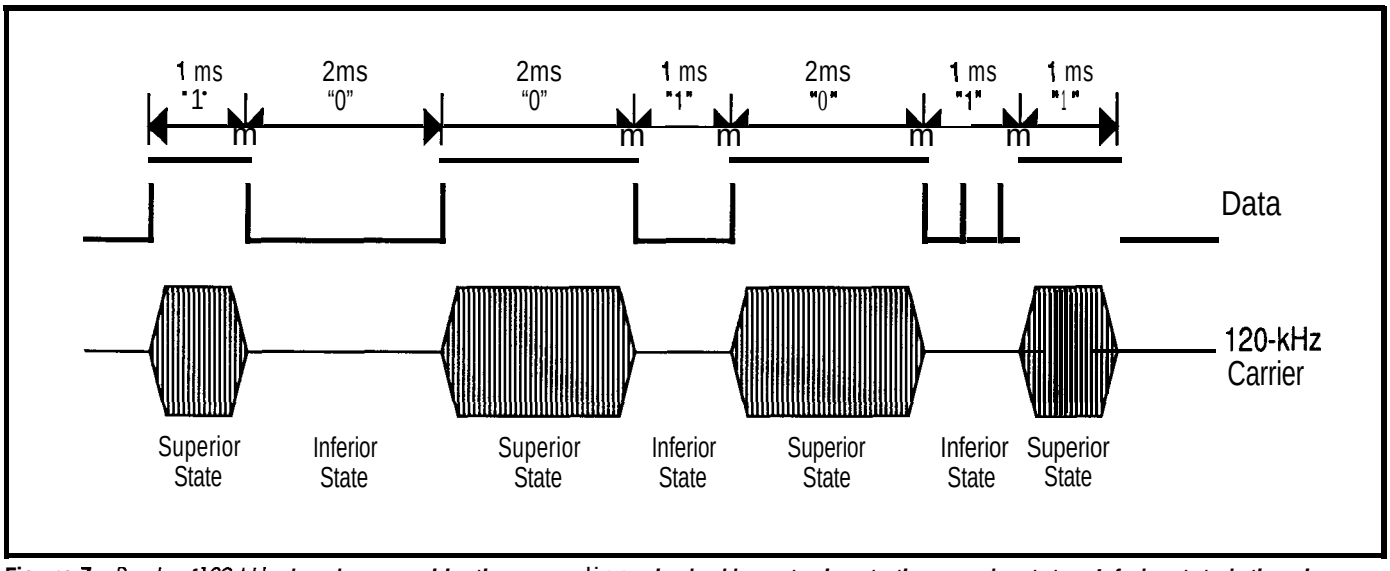


Figure 7—*Bursts of 120-kHz signal are used by the power* line *physical layer to denote the superior state. Inferior state is the absence of signal. Symbols are encoded* in *terms of state duration.*

methods difficult. A fairly simple and low-cost power line communication method was chosen by the committee for CEBus, but it trades off a good amount of speed in exchange for those features.

PLBus uses bursts of 120-kHz signal, known as the "superior state," to send bits of information, similar to the way X-10 works, but asynchronous to AC zero crossings (in fact, PLBus will work without any AC on the line as long as the transmitting and receiving devices are getting their power elsewhere). An "inferior state" is denoted by the lack of 120-kHz signal.

PLBus uses a unit symbol time of 1 ms, so a "1" is 1 ms long, a "0" is 2 ms long, an EOF is 3 ms long, and an EOP is 4 ms long. As such, the theoretical throughput is 1000 "1"-bits per second. Since the other symbols are longer, maximum throughput is less, but the channel speed is typically referenced in terms of "1" bits per second.

Bits are sent by alternating between superior and inferior states, with the duration of each state determining which symbol is being sent. For example, Figure 7 shows that a "0" bit may be represented by either the superior state or the inferior state.

"What about X-lo?" you might ask. "Won't the 120-kHz signals used by CEBus and X-10 conflict with each other?" The answer is a qualified "Yes." It is possible for a valid CEBus packet to look like a valid X-10 transmission to X-10 modules under rare conditions. Since CEBus is so much more sophisticated, it will discard any received X-10 transmissions as line noise and retransmit any resulting garbled CEBus packets. Since early adopters of CEBus devices will likely be consumers who have a small investment in X-10 devices and won't want to just throw them away, it would be bad policy for CEBus devices to cause problems with X-10 devices. Enter the null symbol.

X-10 transmissionsdenote "l"bitswitha 1-msburst of 120-kHz signal and "0" bits with a lack of that signal. One bit is transmitted at each zero crossing, or once every 8.3 ms. Messages are made up of a unique 4-bit start code, followed by 9 bits of data with each data bit sent in its true and complemented form, so a complete message is 22 bits, or 183.3 ms long. If a CEBus packet contains a valid start code (1110) that happens to line up with the power line zero crossings, the eighteen bits following that are going to be interpreted by an X-10 module as a command. If those bits happen to make a valid X-10 command, the module will trigger falsely.

If a rule can be enforced on CEBus transmissions such that no transmission can ever be interpreted as a valid X-10 command, false triggers will be eliminated. The zero in the start code plus eight command bits last 158 ms. If a silent period lasting 15 X-10 bit times (125 ms) is inserted at least every 158 ms, false triggers can be eliminated. As a result, the CEBus specification for the power line's PLSE layer dictates that if there has been 158 ms of continuous channel activity, a 125-ms null symbol must be inserted. The null symbol is removed by the PLSE on the receiving end so the Data Link Layer never sees it.

The bad news, besides complicating the hardware and/or software necessary to implement the interface, is the performance hit. Over 44% of the channel bandwidth is wasted on null symbols, lowering the maximum potential throughput to under 560 "1" bits per second.

As something of a peace token, the spec also makes provision for a null symbol "switch." While manufacturers must build all transmitters and receivers to handle the null symbol, they **may** also include a user-accessible switch that disables use of the null symbol during transmissions. In installations where no X-10 devices are being used and performance needs to be maximized, the null symbol may be discarded altogether.

While we can understand the use of well-established technology for the power line communication, faster and more reliable methods do exist. Perhaps a better method will be suggested during the comment period that will be enough of an improvement to prompt the committee to supplement or replace the proposed method.

TWISTED PAIR

The next physical layer likely to be released is TP. The TP working group is currently testing a number of modulation schemes for performance and cross-channel interference. Current thinking has a full-blown TP implementation consisting of four pairs of wires, with one pairbeing acombinedcontrol/datachannel (TP0) and the other three data channels (TP1–3). Each of the data channels would be broken into several fixed-bandwidth subchannels. A device that wants to use, say, 20 kHz of one of the data channels would request that much bandwidth. If each subchannel was, say, 10 kHz wide, the device would request two adjoining subchannels. The exact channelization scheme is still being worked out, however.

Other issues still to be resolved include the physical connector, whether to allow devices to support fewer than four twisted pairs,and how to specify wiringaspectsin the standard.

INFRARED

Close on the heels of TP is IR. The IR working group is also conducting testing to determine how to proceed with the specification. At present, they are shooting for a data rate of 10,000 "1" bits per second, with a carrier frequency higher than the de facto industry-standard 40 kHz, perhaps somewhere in the 70–80-kHz range.

There is talk of having two types of service denoted as "Level 1" and "Level 2." Level 1 service, also known as "blaster" service, would be a transmit-only device similar to today's hand-held remotes. Such a device would have the advantage of being relatively cheap and easy to implement, but it wouldn't be able to detect other devices transmitting at the same time and wouldn't conform to the CSMA/CD protocol. Level 2 service would be bidirectional service and would conform to all CEBus channel access conventions.

COAX

CX is still in the early stages of development, so there isn't much to report. There will likely be two coax cables involved, one called the upstreamcable and one called the downstream cable. A supervisor **node, Node 0, would be used to coordinate** the control channel and direct video onto the correct cables. With the bulkiness of coax cable and the expense of retrofitting existing construction, the demand for a coax standard will likely be less than for the other media. A draft standard is still a ways down the road.

RADIO FREQUENCY

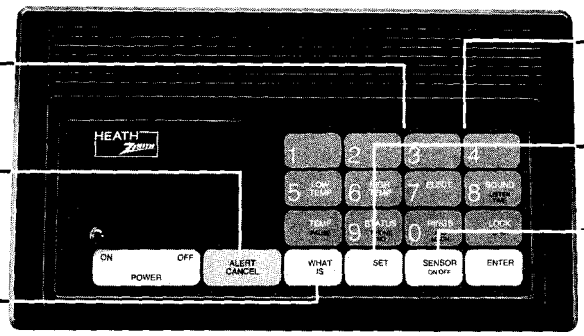
The CEBus committee recently sent out forms asking companies to propose RF signalling methods for possible

THE HOUSE SITTER THAT DOESN'T NEED A KEY

Use *the keyboard to set emergency phone numbers, high and low temperatures, listen-in time and more*

ALERT/CANCEL key cancels automatic dial-out, allows you to answer phone

WHAT IS key lets you listen to function settings and dial-out numbers



Use the keyboard to ask for information

SET key allows you to change previous settings

SENSOR ONIOFF key chooses the functions

-to report

ONLY $129.95

"This is 555-3210. Alert condition is OK. Temperature is **65".** Electricity is on. Sound level is OK." Monitoring your home from work or a vacation spot is made easy with the Heath/Zenith House Sitter Security Monitor/Dialer.

Monitors Your Home
When you call, the House Sitter will report on the AC electric power, the room temperature – comparing it with high and low limits you've already set, loud noises such as burglar alarms and fire alarms , the unit's own battery backup condition, and an additional alert condition. You can even listen to the sounds in the room using the built in microphone.

©1990, Heath Company
Subsidiary of Bull Data Systems, Inc.

Dials Out In Alert Conditions
Set the unit to call out to your office, neighbors' and relatives' to announce any alert conditions that are outside preset limits. Up to four numbers can be programmed.

Order Toll Free l-800-253-0570
The SD-6230 House Sitter is yours for only $129.95*. To order, call toll-free I-800-253-0570. VISA, Mastercard, American Express or your Heath Revolving Charge card accepted. Use order code 620-XXX.

See our full line of electronic products for your home in the Heathkit catalog – call 1-800-44-HEATH for your FREE copy.

Heath®
Benton Harbor, MI

*Price does not include shipping and handling, or applicable sales tax

Reader Service #181

use by CEBus. During the early stages of power line development, such a request was answered by General Electric who proposed their HomeNet protocol. HomeNet was later used as the basis for the present power line specification. Anyone interested in proposing an RF signalling scheme for use by CEBus should contact EIA as soon as possible.

WHERE IS HOME CONTROL HEADED?

Critics may complain that the only way home control is ever going to catch on is if it's simple enough for Joe American to walk into his local K Mart, buy the pieces, bring them home, and plug them in. Early proponents of CEBus promised that having a standard would make all installations, no matter how complicated, just that simple. While it may be as simple as that for small, uncomplicated systems, larger, more elaborate systems are still going to need a professional installer to be done right. CEBus will make the components compatible across manufacturer lines, but it won't work miracles.

Look at today's alarm market, which nobody would deny has shown remarkable growth in the last few years, for a good analogy. A simple alarm system can be purchased at Radio Shack and installed by most home owners with minimal effort. Even simpler are the wireless systems beginning to proliferate into many discount stores. However, for a more complex (and reliable) alarm system, the professional must still be called in. The products the professional uses probably come from different manufacturers, but they are made to work together. You still need someone who knows what he's doing to come in and make everything work in harmony.

The up-and-coming home-control market will likely follow in the footsteps of the alarm industry once CEBus-compatible equipment hits the market. These are interesting times in the field of home automation. I'm looking forward to what's to come.

Source

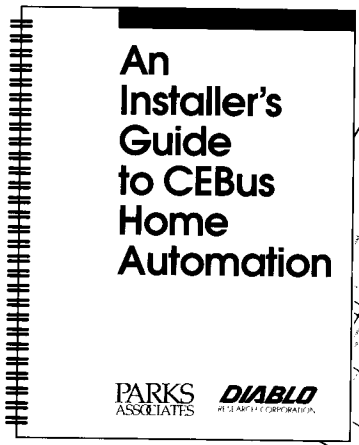
EIA CEBus Proposed Specification
EIA Standards Sales Dept.
1722 Eye St. NW
Washington, DC 20006

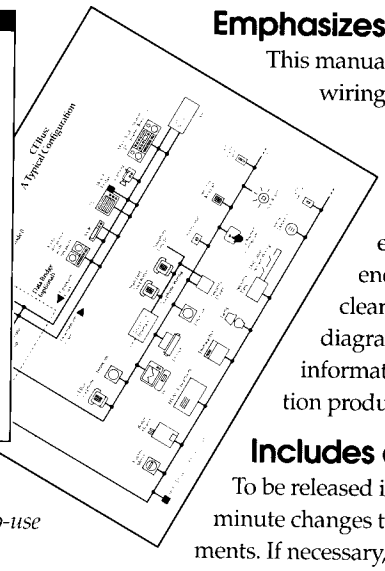
Ken Davidson is the managing editor and a member of the Circuit Cellar INK engineering staff. He holds a B.S. in computer engineering and an M.S. in computer science from Rensselaer Polytechnic Institute. In his spare time, he enjoys making his home into one big automated gadget.

IRS

290 Very Useful
291 Moderately Useful
292 Not Useful

Hands-on info for CEBus automation.





The manual contains easy-to-use instructions, including graphics and diagrams.

Emphasizes the new CEBus™ standard!
This manual provides detailed instruction on the backbone wiring that will interconnect the electronic home of the 90s.

For installers of all types, and for all applications.
Emphasizes CEBus and its application for security, entertainment, lighting, telecommunications, and energy management. Designed for on-site use, with clear, easy-to-use instructions, including graphics and diagrams. Written by Diablo Research, it reveals "insider" information on how to wire for current and future automation products and services.

Includes a free update.
To be released in late 3Q 1990, a free update will include the last minute changes to the CEBus standard and other new developments. If necessary, a later update will be provided.

Order the first edition today.
An Installer's Guide to CEBus Home Automation (delivery April 1990) is available for $149.00, plus shipping & handling. To order, call Parks Associates at (214) 369-5581, fax (214) 369-5582.

PARKS ASSOCIATES

© 1990 Parks Associates. CEBus is a trademark of the Electronic Industries Association.

Reader Service #189

FEATURE ARTICLE

Build a Low-Power Data Logger

Steve Ciarcia

Computerized Data Collector Runs For Years on a Battery

Ever design something so well that it no longer served the purpose that it was originally designed for? I know it sounds absurd but I have to apologize for doing exactly that. Let me explain.

As most of you know, I lead a fairly reclusive lifestyle. By Connecticut-shopping-mall and condos-everywhere standards, we rustic upstate Yankees live in the boonies among the trees. Someone from Montana would laugh at what we call woods, but a New York City native might think he was on a wilderness trek visiting our part of Connecticut. It is all relative, of course.

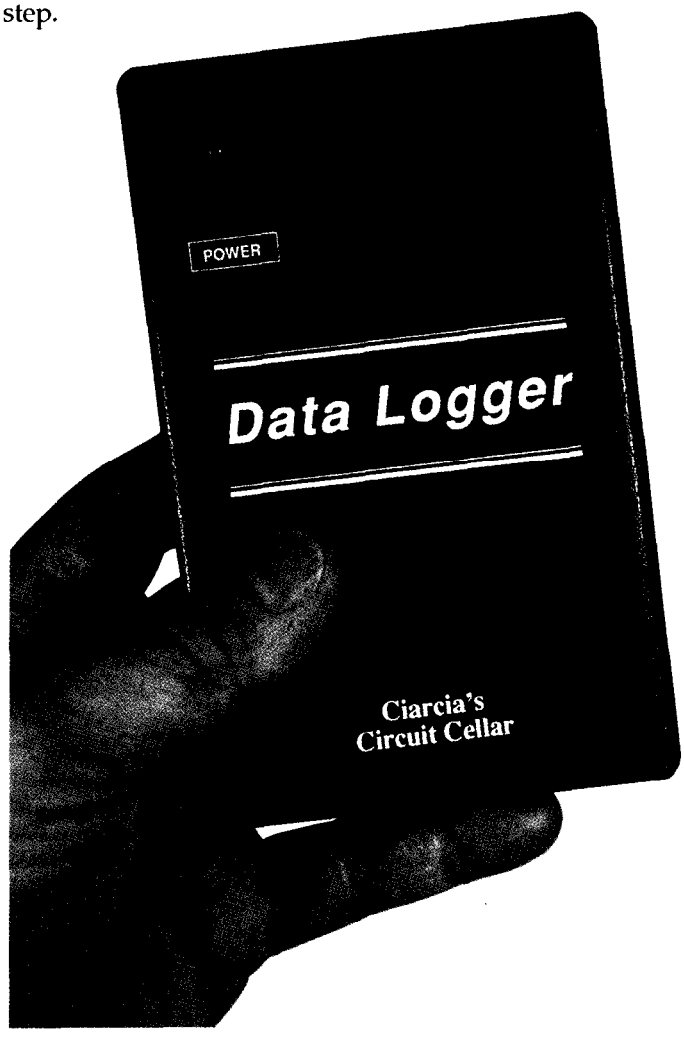
While I live in a wooded area, about the only thing around here that's really rustic are the trees. Our house is a California Redwood hexagonal contemporary that exemplifies the personal style and expression of a mad scientist with carpentry tools. Think of a wooden octopus and you have an accurate description of the ground plan. Next to traditional New England architecture it looks strange, but to me, it's the Circuit Cellar.

Being an engineer whose expertise is process control has left its mark. Our home contains about as much copper as wood. There are wires going everywhere. The security system talks to the home control system (HCS); the driveway sensors (described in issue #14) talk to the video and security system; the video system talks to the home control system; the entertainment system talks to the lighting control system; and so on. Unfortunately, only the environmental control system talks to nothing.

I've been able to install centralized computer controls on everything except the heating and air-conditioning (HVAC) system. It's not that I can't instrument them, it's that I've had relatively little success in proving enough tangible benefit to justify the conversion.

One time I thought I could improve on a mechanical thermostat. I wired a temperature sensor from the master bedroom to the HCS and allowed the HCS to control the AC power to the air conditioner. In combination with the knowledge gained from other sensors, like outside temperature, I presumed I could calculate heating and cooling ramps and anticipate demand more efficiently. Unfortunately, the wonderful shade trees next to the house flattened out the cooling demand ramp while the oversized equator-use-only sale air conditioner I had took about a half hour for a 30" drop. (In the back of my mind I always worried that a little computer glitch could turn the place into a meat locker.)

Fancy monitoring and control algorithms were useless. Other than allowing me to automatically disable the air conditioner while away on trips (about as effective as pulling the plug), the end result was that the HCS merely simulated the simple bimetallic thermostat that was already there (on **when** it's hot, off when it's cold). Computer overkill. Now I leave the power on and let the air conditioner thermostat do its own thing. No use sleeping in a hot room or waking up with frostbite because the Home Control System got trashed by a bad instruction step.



COLLECTING DATA BUILDS UNDERSTANDING

Unlike the black-or-white decision logic of security and lighting controls (needing to see in the dark, for example), HVAC modifications deal with subtle improvements in regulation and efficiency. In some cases the potential improvements don't warrant the expense of elaborate controls just so someone can say a house is computerized. Houses that are designed from the ground up with solar and HVAC efficiency in mind are better served by computer controls because they frequently contain many more controllable functions like vents, flow controls, fan speeds, automatic window shades, and multisource heat storage and distribution systems. Successfully retrofitting traditional architectures with elaborate energy controls in hopes of obtaining equivalent efficiency is less clear-cut and may not be cost effective.

Deciding whether or not you should blast holes in the walls for automatic exhaust fans or install an auxiliary 25,000-BTU gas heater to even out demand peaks takes more thought before swinging the sledge hammer.

I learn from my mistakes—My sledgehammer is holstered.

Fortunately, I learn from my mistakes, and my sledge hammer is presently holstered. Last fall we added a solarium and sunroom-style greenhouse with the intent of making it energy efficient, perhaps even contributing heat to the rest of the house. It also has a wood stove.

During its construction I strung wires through the walls and rafters for all the controls and instrumentation that could be needed. Want an LCD panel on the wall in the solarium that displays present conditions and a histogram of control activities? The wires are there. Want to pick up the smoke stack, wood stove, and greenhouse glass temperatures? The wires are there. Want to pull heated air from the solarium into the house? The fans are there.

The idea behind all this monitor and control wiring is to efficiently regulate the temperature of the solarium and channel any extra heat into the central section of the house. Rather than immediately design a controller, however, this time I decided to collect data for a period of time to see how much control was actually necessary. Did the sun only shine in the greenhouse one hour a day and provide no heat gain? Did the solarium temperature drop to the chilly outside air temperature the instant the wood fire died down? Was the 450 square-feet of glass in the solarium actually a net heat loss? Should the fan direction be changed (ugh!) to provide heat to it from the house? Certainly these questions had to be answered.

DATA LOGGING BASICS

Recording the inside, outside, floor, stack, firebox, and house temperatures is no problem. Got an alarm clock and a lot of vacation time stored up?

Obviously, we could hang a lot of thermometersaround and sit in a chair with an alarm clock and a pad, but that sounds like a real waste of time. My first inclination was to drag one of the spare PCs up from the Circuit Cellar and sit it in the middle of the solarium. Using off-the-shelf ADC and parallel I/O cards, I could easily instrument the entire room, take readings on demand, and save the data to disk.

I dismissed the thought almost immediately. While general-purpose computers like ATs can be configured to record analog data, we are limited in how we use them because of their generalized architecture and tremendous appetites for electricity. I can record 10,000 temperature readings a minute with an AT and record all this useless data to a 160-Mbyte hard disk. However, when the power goes out, how long does the UPS last? Twenty minutes? One hour? Did the power go out during the kind of weather I really *want* to monitor?

What we have to do first is understand that there is a fundamental difference between acquiring data and the control and data analysis that we typically associate with data acquisition systems. Acquiring data is an input-specific function; analysis is a processor-specific function; and control is an output- and processor-specific function. The latter two consume the majority of system power.

If we wereable to separate the tasks then we would not be burdened with the expense or power consumption of unneeded functions. For example, wouldn't it seem logical topower the input section only while it acquired data, the processor only when it was processing the data, and the output only when it was reporting results?

In point of fact, computer systems like this do exist but are generally used in avionic applications. Instead, my example is meant to suggest the necessity for a specialized low-power device whose sole purpose is to record data which can later be communicated to a computer for analysis or display. Rather thanremotely wire the world to central computers, remotely log the data and bring it back to the computer.

Coincidentally, the device that performs this function is called a data logger. Data loggers come in a variety of configurations but their common attribute is that they are generally self powered and designed to record data for long periods of time.

Data loggers may ormaynot contain microprocessors. For recording a single 8-bit value at a predetermined interval, only a memory chip and simple CMOS counter circuit are necessary. Multiple readings, calculated intervals, or preprocessing (ignoring bad readings, for example) of data as it is recorded requires an integral processor. How much this simple data logger starts to resemble a general-purpose computer determines its power efficiency and potential application.

Power conservation is what separates a true data logger from the general-purpose computer performing similar activities. The fundamental operation of a data logger counts on the fact that it is almost always off. A data logger only turns on when it has to log a reading (determined by external interrupt or time interval) and then turns itself off again.

To further conserve power, the timing, input, and processor/memory sections of the typical data logger are often independently powered. A very low consumption continuously powered timer periodically wakes up the processor and I/O by turning their power on as needed. At the end of the logging sequence the computer resets the timer and shuts itself off until the next event.

THE CIRCUIT CELLAR DATA LOGGER

The major design premise of my data logger was portability and programmability. Today I needed it to measure temperatures in the solarium but tomorrow I might want to take it out to monitor the pH and temperatures of a brook with some suspect upstream neighbors. Consider the bombshell when you document a sudden pH drop at about 3 A.M. every Monday morning.

The Circuit Cellar Data Logger is actually an application-specific microcontroller designed around the CMOS 80C52-BASIC processor. I chose this processorbecauseit's CMOS, it has a built-in BASIC interpreter (I don't like programming anymore than I have to), and any program I write will be changed by you anyway, so why not make it easy to do? If you want to lower the cost of the project, simply use an 80C31 processor and assembly language.

As previously mentioned, a data logger is basically a battery-powered timer, processor, and ADC. The uniqueness of a data logger is primarily its timing section: the circuit that knows when to wake up the system. Of course, predetermining an interval timing range that meets all applications takes a crystal ball.

I actually designed two different timing sections that could control the same basic processor and I/O sections. The first was an attempt to be everything to all people by providing a programmable interval that could be changed at each reading. With this circuit, intervals as short as 1/64 second or as long as 256 hours between samples could be

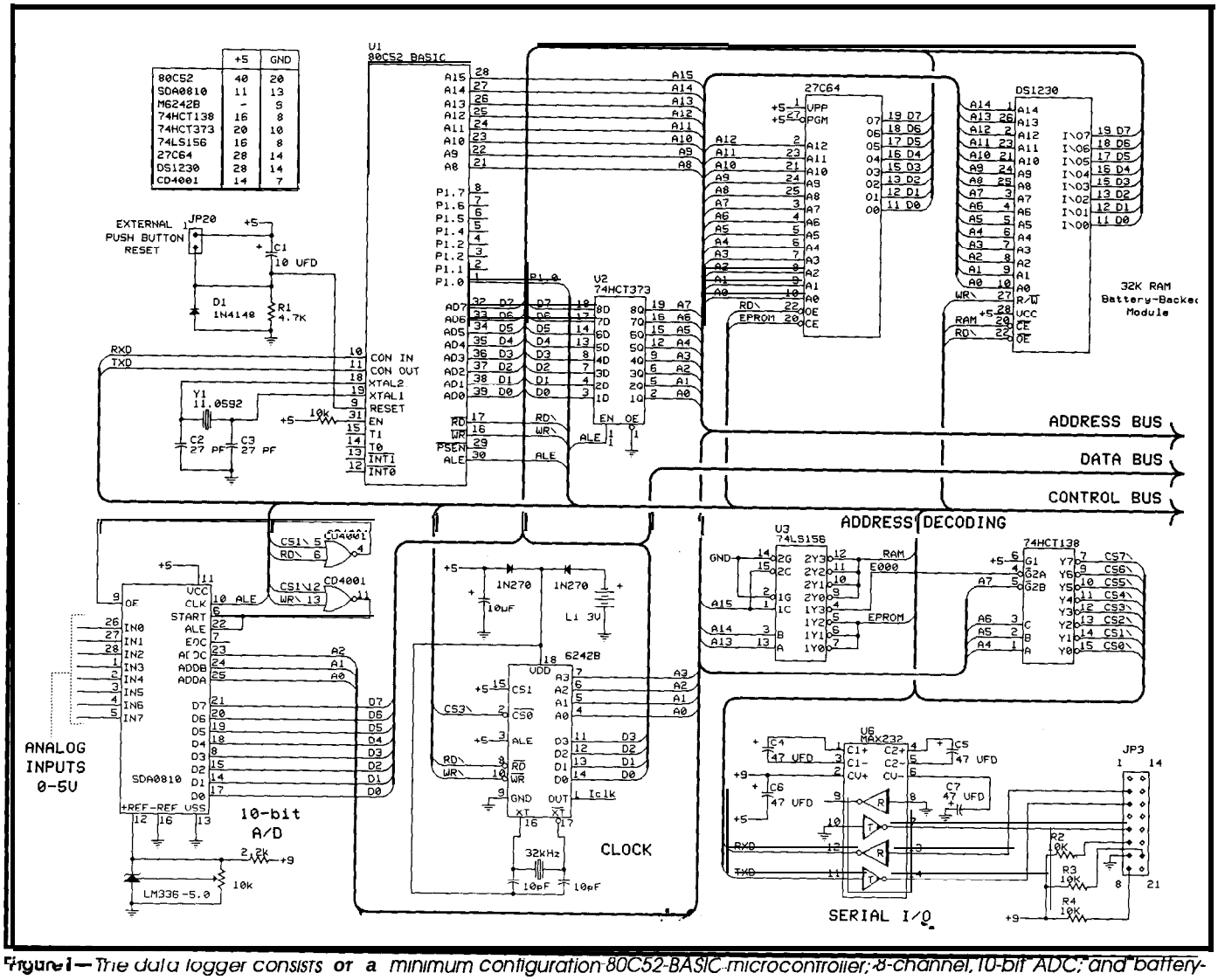
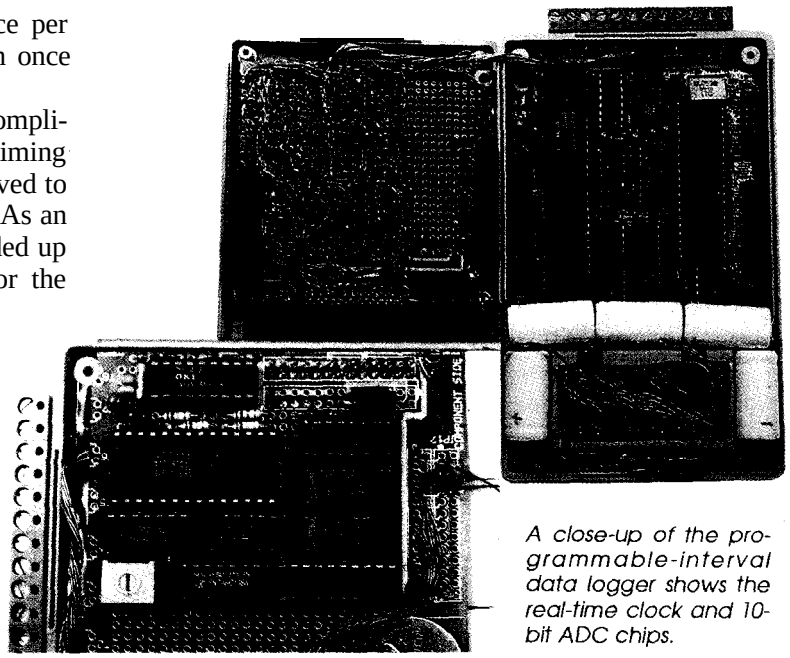


Figure 1—The data logger consists of a minimum configuration 80C52-BASIC microcontroller, 8-channel, 10-bit ADC, and battery-backed real-time clock.

set automatically. Or, samples could be taken once per minute until they reached a specific value and then once every three seconds for example.

The second fixed-interval timer was much less complicated but also less flexible. Offering 12 selectable timing intervals between two seconds and two hours, it proved to be more than adequate for my solarium application. As an added benefit, its simple power control circuitry ended up functioning as an uninterruptable power supply for the whole system. It could even be solar powered. More later.



A close-up of the programmable-interval data logger shows the real-time clock and 10-bit ADC chips.

THE PROCESSOR SECTION

The processor section of the data logger is a minimum configuration 80C52-BASIC microcontroller (basically a stripped down RTC31/52) shown in Figure 1. ***[Editor's Note: See*** *"From* ***the Bench"*** *in issue #8 of* CIRCUIT CELLAR *INK for diagrams* and discussion of the *RTC31/52.]* It has two memory chips, a 32K-byte RAM addressed at 0000H, and an 8K-byte CMOS EPROM addressed at 8000H. The EPROM holds the autostart BASIC program that will automatically execute when power is applied.

The RAM in this system is unique because it is combined with a Dallas Semiconductor DS1213C battery-backed RAM socket to provide nonvolatile system RAM (you could also use a DS1216C SmartWatch socket). No use logging data that gets erased when the power goes off.

The RAM chip is a special low-power device that can be either an 8K-byte (Toshiba TC5565) or 32K-byte (Sony CXK58255P) chip depending upon how much data has to be stored. Data can be retained for up to 10 years in one of these sockets.

Before we can take advantage of nonvolatile RAM in a BASIC-52 system, we have to be aware of its idiosyncrasies. When BASIC-52 powers up, it automatically clears RAM. However, if the program stored in EPROM is saved with a PROG4 command rather than simply PROG, the system will erase memory only up to MTOP, the top of memory. If, before we program this EPROM, we move MTOP down to perhaps 2000 (decimal), a PROG4 will retain this value and we have 2K bytes for variables and program stack and 30K bytes available to log data.

In the simple data **logging** program that I wrote for the solarium, a PRINT FREE statement came back with 32104 bytes. The 2K that I had allocated was more than enough. It was using barely 700 bytes. If you needed more than 30K or so of RAM then the next thing to do is replace the EPROM with another 32K-byte nonvolatile RAM socket. While this circuit does not support the EPROM programming features of BASIC-52, we can simulate them if there is a RAM in the EPROM socket. Using the program in Listing 1 appended to your data logger program, do a GOTO 10000 and it will simulate all the PROGx commands. Since the processor does not erase memory at 8000H when it starts, the program will reside and execute as if it were an EPROM. Program changes can be easily made with a terminal and "PROGed" instantly. My prototype actually used two nonvolatile RAMs even though schematically they are designated as a RAM and EPROM.

The final part of the basic controller circuit is a MAX232 which serves a dual purpose. When the system is powered it allows serial communication between the data logger and a host computer or terminal. The readings are most easily dumped serially but it can also serve to send serial strings whenever readings are being taken. In some cases you may want a hard copy of specific events so a serial connection to a companion printer is in order.

Before you pull out the MAX232 to save power because you don't need any serial communications, realize that the MAX232 is also a pretty efficient DC-to-DC converter. In the data logger, the MAX232 also supplies +9 V to the ADC reference.

A REAL-TIME NECESSITY

There is a microprocessor in the control section of a data logger because it is usually the most cost effective means for performing the task of reading and recording data. The accuracy and timing of the data are more significant, however. Whether the data logger is of fixed or programmable interval type, it is not enough to simply count intervals to determine absolute real time. Especially on systems that allow external inputs to trigger data logging, a crystal-controlled real-time clockmust be included so the event time can be recorded as well.

Figure 1 also details the circuit for an OKI6264B battery-backed clock/calender. Operating on a 32,768-Hz crystal and a 3-V lithium battery, the real-time-clock is set or read directly in BCD digits. This makes it easy to use in BASIC. As configured, it occupies addresses E030H to E03FH. In addition to clock/calender functions, the 6264B can be programmed to output a square-wave clock signal

with periods of 1/64 second, 1 second, 1 minute, or 1 hour. The basic command XBY(0E03EH)=4 will set thisclock to seconds while XBY (OEO 3EH)=12 sets it to hours.

A-TO-D CONVERSION IS A RELATIVE QUESTION

The singular component responsible for the accuracy and application of a data logger is the ADC. Accuracy, like the weather, is relative.

Anyone familiar with my editorials will remember a tirade where I criticized people who think that the only valid data extends to nine significant digits past the decimal point. Perhaps in military circles where price is no object this assumption has merit, but here in the real world, analog acquisition accuracy translates to the same number of digits past the dollar sign. Designing for the accuracy you need, rather than for the accuracy **you** think **you** want, results in a more cost effective design.

Yes, I can use a 16-bit ADC to read the temperature values in the solarium, but I wonder how much of any reading is valid. Unless the ADC section is built on a multilayer PC board, is shielded from EM1 with a metal enclosure over the ADC section, has separate power and grounds for the analog and digital sections, and has lots of input filtering, a significant portion of any reading may be noise. For a 1-V full scale, a 16-bit converter has a resolution of 15 μV! Put a scope on the average single-board microcontroller with ADC and you'll measure a few mV of electrical noise just from the processor. You may start with 16 bits, but after filtering out the noise from sensor wiring and the processor, the effective accuracy will probably be more like 10 or 12 bits. There is a place for real 16-bit or greater accuracies, but it's not here.

```
9999  END
10000 PRINT "Hit P to move your basic program
      to RAM at 8000H"
10010 PRINT "(Hit Q to quit without executing
      prog command)"
10012 PRINT "(Hit C to clear NVRAM at 8000H)"
10015 G=GET
10020 G=GET : IF G=0 THEN 10020
10022 IF G=43H THEN GOSUB 10200 : GOT0 10000
10025 IF G=63H THEN GOSUB 10200 : GOT0 10000
10030 IF (G<>50H.AND.G<>70H) THEN END
10040 XBY(8010H)=55H
10050 FOR X=200H TO (200H+LEN)
10060 XBY(X+7E11H)=XBY(X)
10070 NEXT X
10080 PRINT "Hit l-6 to do a PROG# to RAM
      at 8000H"
10090 PRINT "(Hit Q to quit without executing
      PROGx command)"
10095 G=GET
10100 G=GET : IF G=0 THEN 10100
10110 IF (G<31H.OR.G>36H) THEN END
10120 XBY(8000H)=G
10130 XBY(8001H)=INT(RCAP2/256)
10135 XBY(8002H)=RCAP2-(XBY(8001H)*256)
10140 IF G<32H THEN END
10150 XBY(8003H)=INT(MTOP/256)
10155 XBY(8004H)=MTOP-(XBY(8003H)*256)
10160 END
10200 PRINT : PRINT "Clearing NVRAM": PRINT
10210 FOR N=32768 TO 40960 : XBY(N)=255 : NEXT N
10220 PRINT : RETURN
```

listing 1—*Code to simulate the* **PROGx** *command for battery-backed RAM is tacked on to the end of any user program.*

Environmental conditions change slowly. We live in a world centered around 70°F. The fact that it changes from 70.0041 to 70.0042 degrees may have significance to someone, but it is not what my data logger is designed to record. The reality of life is that it takes changes on the order of a couple degrees in our environment for us to even notice. Furthermore, with the hysteresis built into the typical electromechanical HVAC controllers, the actual temperature may range ±3° about a nominal setpoint. What is the benefit of measuring it out to four decimal places then?

The ADC section of the data logger, also detailed in Figure 1, is designed around an SDA0810 8-channel 10-bit A/D converter. The SDA0810 is pin and function compatible with the venerable ADC0808 from a variety of sources. You can also use the ADC0808 in place of the SDA0810 if you don't need the extra two bits of resolution.

The reference to the ADC chip is set at 5.00 V by an LM336-5.0 reference source. The LM336 is powered by the high-voltage output of the MAX232. One caution here is that different brands of ADC0808s seem to take more or less reference current. The SDA0810 required about 100 μA while at least one Samsung ADC0808 took 2 mA. If you have any other components powered from the MAX232, check that it is capable of supplying all of them or that the series dropping resistor doesn't need to be lowered a bit.

The SDAO810 functions just like the ADC0808. As configured, the ADC is addressed at E010H through E017H. To set the channel to be converted, we first write a dummy value to that channel. To read channel 3, for example, we execute an XBY (0E013H) =O. After the conversion, the most-significant eight bits are read by executing an N1=XBY(0E010H). N1 is the 8-bit ADC value normally obtained if this were the ADC0808 chip. On the SDA0810 the additional two least-significant bits are obtained by doing another read, N2=XBY(0E010H), from the ADC before starting another conversion. The 10-bit value then becomes the logical combination of N1 and N2.

The SDA0810 is set for a range of O-5 V. While I could have optimized this range for finer resolution from the temperature sensors, I have other sensors on the drawing boards for another application that will use the entire range. As it stands, the temperature sensors put out 10 mV/°F. A 10-bit ADC with 5-volt range has a resolution of 5 mV per bit. I think 0.5" accuracy is fine for my application. If you need more detailed readings then either amplify the sensor output or reduce the range of the ADC.

Figure 2 illustrates a few sensors which are appropriate for use with this ADC. For measuring temperature I like the LM34 because no value conversion is necessary. If the temperature is 72°, the LM34s output will be 0.72 V. If it is 85" it will be 0.85 V. The other circuits sense light levels and intensity. These indications are relative, of course, and you'll have to experiment to determine relevance.

IT'S ALL IN THE TIMING

Thus far, our data logger consists of processor/mem-

ory, ADC, and a real-time clock. The only thing left to add is "personality."

The heart of a data logger is the low-power timing circuit that turns the processor on at the appropriate time. As I mentioned before, these timers can have fixed or programmable intervals.

Probably the first question you might ask is why we couldn't just use a CMOS one-shot or one of those CMOS 555-based long-interval timers. Well, on the scale of relative power consumption, you could toast bread over the venerable CMOS 555 and its clone cousins. The typical CMOS 555 takes 200 µA or more. That's a lot when it has to come from a battery. CMOS one-shots aren't particularly accurate for long durations and are just about as powerhungry.Theexcessivepowerisbecausethesedevices are operating **in a linear mode** when they are timing.

CMOS digital logic is extremely efficient and, as long as we don't operate it in a linear range between Vcc and ground, power consumption is essentially zero. Translated, this means that CMOS logic only consumes power when it makes a logic level transition. Any other time it merely consumes its quiescent power which is typically 1 µA at 25°C. The best method to create accurate timing and still have very low power consumption is to clock a series of CMOS counters. The slower the clock frequency, the fewer transitions and power required. Now you know why I didn't start with a 3.58-MHz crystal.

Figure 3 details the circuit of a programmable interval timer. It consists of an B-bit programmable divide-by-N counter whose input comes from the real-time clock's interrupt output. An B-bit register holds the BCD divisor value. Executing an XBY (0E000H) =5 will set N=5. If the 6264B real-time clock has been set to a cyclic output with a period of one second on its interrupt output with an XBY(0E 0 3EH) =4, then the divide-by-N counter will have an output frequency with a period of five seconds. If the real-time clock were set for hours instead of seconds then

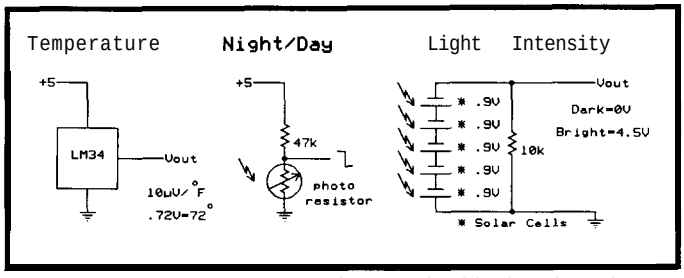
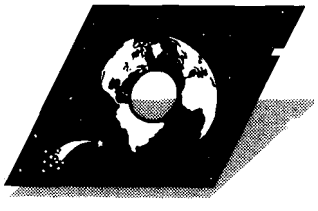


Figure P-Various *sensors can be used with the data logger* **including** *those to sense temperature and light level.*

this interval would be five hours! The selectable ranges are 1/64 to 255/64 seconds, 1 second to 255 **seconds, 1 minute to 255 minutes, and 1 hour to 255 hours (can't divide by 0).**

When the number of pulses into the divide-by-N counter reaches the value N, the output will toggle. This transition triggers a short-duration one-shot connected to the set side of a set-reset flip-flop. Because the pulse width of the divide-by-N counter is the same as its input fre-

Your world has many *parts,*
Tele™ helps you put it all together..



SIMPLIFIED

Berry Computer presents the Tele Operating System A Multi Tasking MS-DOS COMPATIBLE

Simplifyyourapplication.Ifyouarewriting an application to sense and/or control the real world, you may find it much easier to base it on tasks. A single task only needs to know how to deal with its specific part of the environment. That naturally leads to modular, reliable programs.

The Tele Operating System Toolkit is ideal for providing precise, preemptive multitasking to computer systems based on the Intel 8086 family of microprocessors. Tele allows tasks to be **executed** at precisely specified times and to suspend themselves for specific time periods. The current time of day is available with a precision of better than 1 microsecond.

Tele also provides windows so that you can monitor several tasks at once on the display screen. Tasks write to virtual displays, which are simply areas of main memory and therefore very fast. An internal operating system task copies portions of virtual displays to the physical display screen. The amount of processor time consumed by the display task is limited so that time is not wasted updating the physical display faster than the operator can read it.

The Tele Toolkit is written in C and assembler. All source code is provided, and it may be compiled in all memory models (libraries are provided **in the small and large models** for the Microsoft version 5.1 C compiler).

The Tele Operating System includes all object modules and the source code for MS-DOS emulation which is an external function. The Tele Tool Kit components are only required if you need the source code for the internal functions. Object modules are provided for Microsoft C version 5.1 unless otherwise specified.

CALL NOW TO ORDER
(209) **267-0362**
FAX (209) **267-9246***
*Note new Fax #

Tele Operating System	$100
Tele Tool Kit	
Multitasking **(MT)***	**$ 50**
Windows (WI)*	**$ 40**
File System (FS)	**$ 40**
Tele Operating System and Tool Kit	**$200**
Demonstration Disk	**$ 5**

'**MT** formerly called **SK** **WI** formerly called **CD**

Telephone **support** is *freely* available.

Tele **is available from:**

Crosby Associates
P.O. Box 248
Sutter Creek, California
95685

Visa, Mastercard, American Express & Discover Card accepted.
MS-DOS and OS/2 are trademarks of Microsoft Corporation.

Reader Service #178

quency (possibly as much as an hour), the edge-triggered one-shot effectively isolates the level-triggered set-reset flip-flop from being continuously turned on. When the flip-flop is set, its output turns on a relay that applies power to the processor and ADC sections.

The reset side of the flip-flop uses the other half of the one-shot to isolate it from the rest of the system. To turn the system power off, all the program has to do is read or write something to location E070H. An XBY (0 E 0 7 0 H) =0 fires the reset one-shot and turns off system power until the divide-by-N counter fires the set one-shot again.

Using this circuit, it's possible to change the interval between samples through a program decision. Perhaps ratherthanonceperminute,wenowwanttowaitonehour until the next sample (to do that, we merely output a new value of N to the register before the processor shuts itself off). While a fixed-interval logger set for l-minute samples could collect the same data by not recording the next 59 samples, it would still consume power for the 59 times it had to awaken to make a decision to go back to sleep. Programmable-interval timers are perfect where a combination of short- and long-interval sampling is involved.

A LITTLE BLACK MAGIC

Before we go on to the simpler fixed-interval timer, we should discuss power distribution. Mixing powered and unpowered CMOS logic is not an easy task.

The processor and ADC sections share a common +5-V line. The **memory,** of course, shares the same power but is battery backed. The real-time clock has a +3-V lithium battery. For the interval timer to work, it must also be battery powered. The divide-by-N counter, register, NOR gate, one-shots, and the CD4011 NAND gate are all powered by a 4.8-V NiCd battery.

The last key ingredient is the relay. Its main activity is not only to switch power to the processor. Let me explain.

Logically, there is little problem turning off power to sections of a CMOS circuit if you don't care that the current leaking from the active circuit can far exceed the typical operating current of the active part, or that an unpowered device can appear as a logic low to a powered device.

When you design a timing circuit that should typically take less than 100 µA and the meter reads 2 mA, you might feel like throwing in the towel. But, when you read significant voltages present in the unpowered section at the same time, it can drive you positively nuts!

It took a couple days to track down the latter problem. It turns out that if you leave an RS-232 terminal attached to a MAX232 and turn off power to the data logger (as you might do during testing), the terminal's RS-232 voltages will feed through the MAX232 and float the power line to the processor and the rest of the circuit at about 1.3 V. This does nothing to the unpowered processor and is not harmful at all. But it is just enough voltage to cause current flow in the various connections to the remaining powered CMOS section. The address and I/O buses of the unpowered section all start drawing current from the CMOS timer section. Instead of the 50-µA design objective, l-2 mA of current were being drawn by the timer. Absolutely the pits.

The solution was to physically ground the +5-V line of the unpowered section when it was supposed to be off so there could be no potential between it and ground. The normally closed section of the relay serves that function. The normally open contacts apply power when the flip-

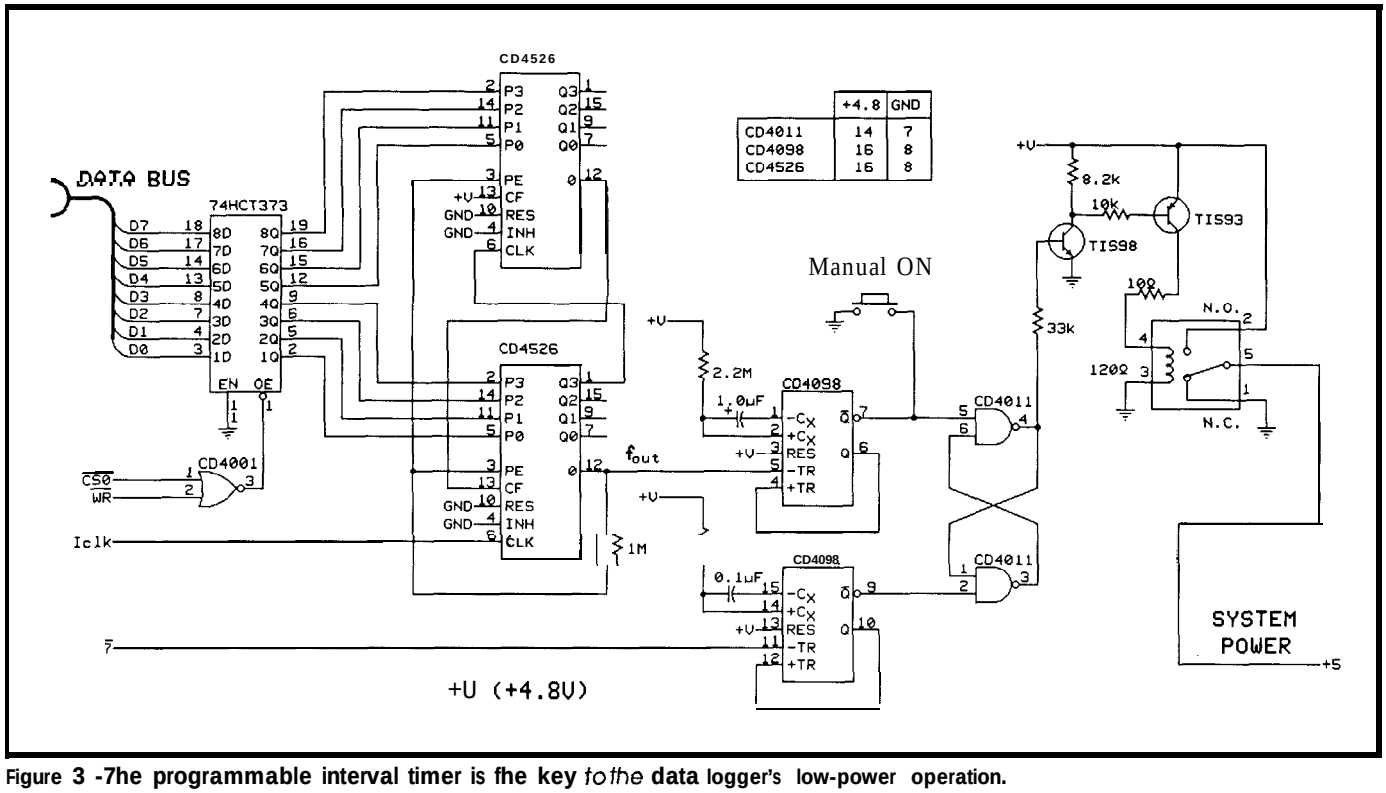


Figure 3 -7he programmable interval timer is fhe key *to the* **data logger's low-power operation.**

flop sets. Once I eliminated all the current drain on the bus connections the circuit worked as I expected. With the real-time clock set for a 1/64-second clock rate, the timing section takes about 40 µA. With the clock set for one-minute periods, it is down to 24 µA. I never did check hours but I would presume it is less again.

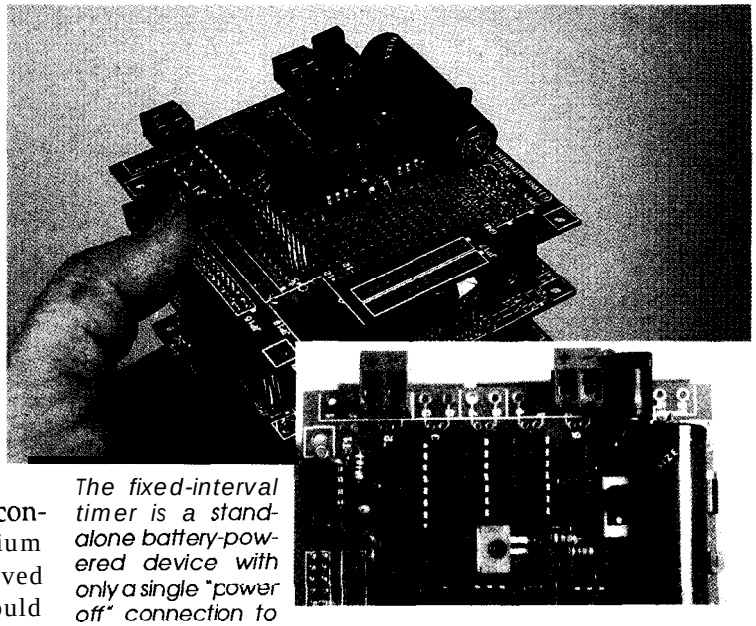
FIXED-INTERVAL TIMING

For all the benefits of the programmable interval timer, it didn't come about without a lot of empirical wisdom. If you don't want to chance a run in with the gods or already know that your crystal ball needs more than polishing, I suggest the far simpler approach of a fixed-interval timer.

Unlike the programmable-interval timer that is directly connected to bus and I/O lines, the fixed-interval timer is a stand-alone battery-powered timing device with only a single "power off" connection to the processor. Once started, the fixed-interval timer triggers an event after the same fixed interval every time.

Figure 4 is the circuit of a very low power fixed-interval data logger trigger. Timing again is crystal controlled and the circuit is powered by one 3-V lithium battery. This time, however, the clock pulse is derived from an MM5369AA oscillator/divider chip (you could also use the output from the OKI6242B as described before). Ordinarily, this chip uses a 3.58-MHz crystal to produce a 60-Hz output frequency. Substituting a 32,768-Hz crystal lowers both the power consumption and the output frequency. Instead of 60 Hz, it is now 0.549 Hz.

This clock frequency is applied to a 12-stage binary counter with the resulting clock period outputs ranging in binary multiples from four seconds to two hours. To select



The fixed-interval timer is a stand-alone battery-powered device with only a single "power off" connection to the processor.

EMBEDDED SYSTEMS DESIGNERS START USING INTEL'S 8096

★★★NOW★★★

COMPLETE DESIGNERS TOOL-KIT UNDER $500.00

MATE97 Single Board Computer- 12MHz 8097 Processor, single voltage supply, RS232 serial support, 50-pin I/O header, 32K EPROM, 32K RAM

MON97 Debugger/Monitor EPROMs—uses terminal or PC, download HEX, assemble, disassemble, step, dump RAM, set breakpoints, over 40 BIOS calls available.

SMALL C-8096 Compiler

MATECOM-Terminal Emulator (IBM-PC)

Complete **Developers Tool-Kit with wall mount power supply and prototyping board $495.00**

8096 Assembler (IBM-PC) $125.00

Other tools available

Call or write today for data sheets and information.

B.G. Associates
P.O. Box 66 • Bowie, MD 20715-0066
(301) 509-6748

Reader Service # 179

CARVING OUT A NEW ERA



- Evaluation kits, RS232 Serial to Power-line Interface, PC Support Software
- I/O and Interface Modules
- Analog Monitoring Modules

ARCATRON
3641 West Phelps Road - Phoenix, Arizona 85023
(602) 843-2589

Reader Service #177

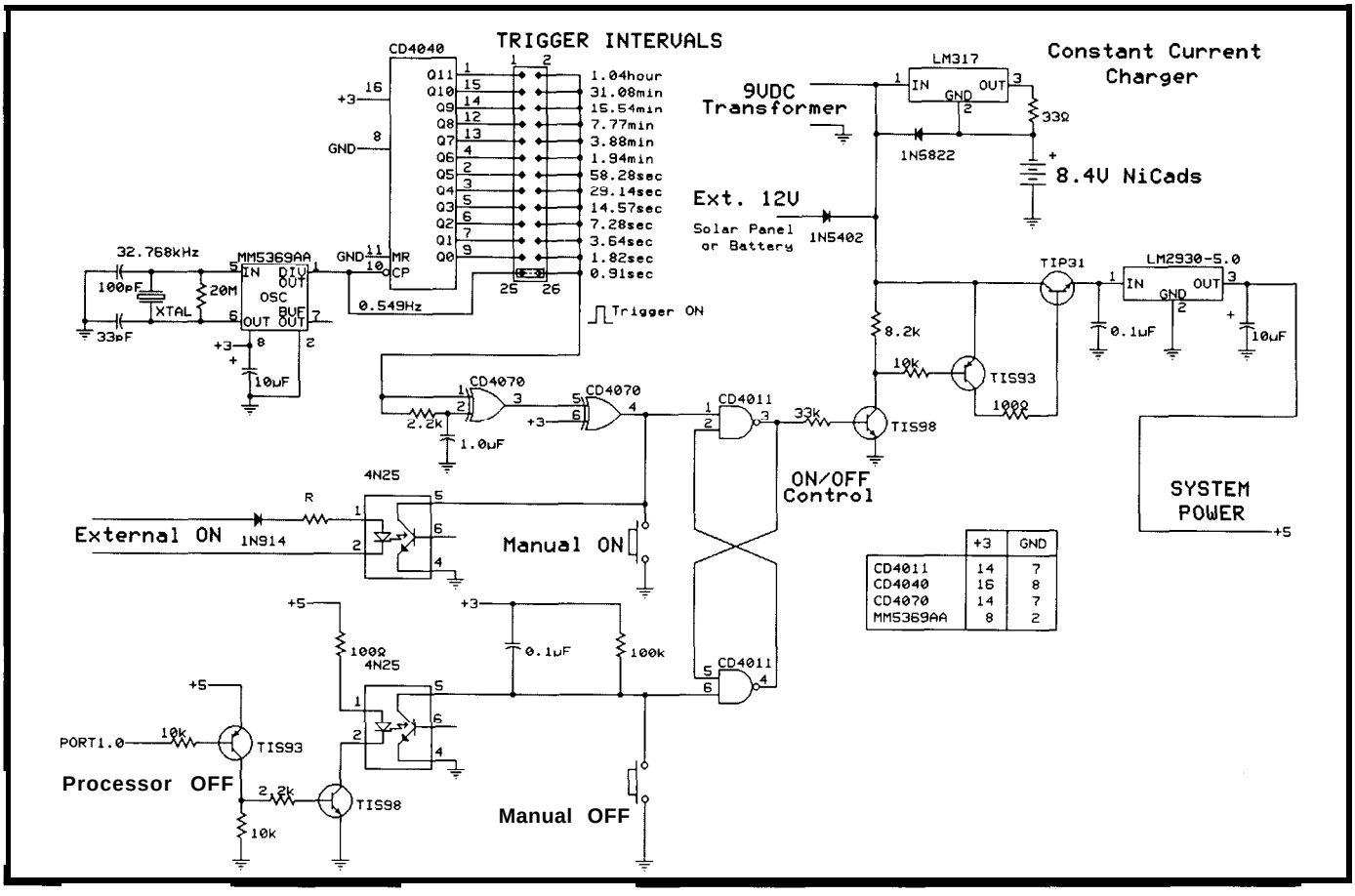


Figure 4 -*A* **simpler circuit than the programmable interval timer is this very low power fixed-interval trigger.**

an interval, we jumper-select an appropriate counter output line. The selected frequency is converted to an edge-triggered pulse so that the set-reset flip-flop will not stay continually set.

The result of using edge-detector triggering is that the time interval will actually be defined as the time between transitions rather than the period of the waveform. Since there will be a transition every half period, the interval will actually be half the period of the measured frequency. For example, while the Q1 output of the ripple counter will have a frequency of 0.549/2 or 0.2745 Hz (3.642-second period), a transition will occur every 1.82 seconds. Selecting Q1 sets a 1.82-second interval trigger time. Selecting Q5 selects a 29.14-second interval. The maximum interval time is 62.17 minutes.

A UPS IN DISGUISE

In addition to on/off control, the fixed-interval timer's power-control circuit incorporates other features including the ability to use a variety of voltage sources. Power can be supplied through a 9–12-VDC modular power transformer, an external 12-V battery or 12-V solar panel, or an internal 8.4-V NiCd battery pack. Whenever an external source is applied, a constant-current-configured LM317 provides 40 mA of charging current to recharge the internal batteries. Once turned on, this source voltage is regulated to +5 V through a LM2930-5 low-dropout, low-quiescent-current 5-volt regulator.

The final connection is the automatic power off from the processor. To eliminate the potential problems in connecting CMOS directly to an unpowered circuit, I used an optoisolator. A two-transistor circuit is configured so that only a hard logic low, not an open or unpowered level, will fire the optoisolator and reset the flip-flop. This reset circuit is connected to bit 0 of PORT1 on the processor. An additional optoisolator isconnected in parallel with the set input to the flip-flop so that an externally applied voltage could turn the processor on as well.

Reviewing the circuit of the fixed interval unit suggests that I had more than one idea in mind. In fact, it functions as a complete automatic/manually controlled power supply for a small computer. If you do not select any jumpers on the interval counter, then using the manual on/off will control the power. Since they are all wired in parallel, the external interrupt "on" or the PORT1 "off" signal, or any combination of them, can also control the power.

Rather than reinvent the wheel and build the second whole data logger from scratch, I used off-the-shelf RTC52 and RTCIO boards populated only with the necessary components. With the battery power supply and timer circuit neatly built on an RTC prototyping board, theentire system now functions either as a data logger using the

timer, or a conventional battery-powered uninterruptable supply (UPS) using the manual controls.

HOW LONG CAN WE RUN THE DATA LOGGER?

As I mentioned, the main attribute of a data logger is that it is almost never on. Only the timer circuit constantly consumes power and it is specially designed to do very little of that. The processor and ADC also draw power only when they are activated.

How long a data logger can run is a function of battery and memory capacities. Generally speaking, however, only naive users apply a data logger like this design to collect voluminous **data** which fills all of memory in **a** short period of time. I designed this data logger for more long term limited data recording and count on an intelligent user to manage the amount of data wisely. With that in mind, the data logger life depends solely on the power consumption of the interval timer and the processor.

The processing and timing sections have completely different battery requirements. One section requires low current all the time while the other section requires high current for short periods. Two kinds of batteries are needed.

The timing section runs all the time at the lowest possible voltage (usually 2-3 V) so that it consumes the least power. The current required in the two timer circuits I presented averaged 25-40 µA. If we power the timing circuit with a 160-milliampere-hour lithium battery then the timer will last 267 days. A 500-mAH battery, which is only slightly larger (the AA-sized Li battery pictured on my prototype is rated for 1.1 AH!) lasts 2.28 years!

OK, now that we know we could put a battery on the timer that could last longer than we will, the real test for service life is the power it takes to run the processor.

The 80C52-BASIC processor and memory circuit in its bare bones configuration takes about 30 mA. Adding the ADC, MAX232, real-time clock, and other logic, brings it up to about 70 mA. The relay in the programmable interval timer adds another 30 mA if you chose to use that circuit.

What this all means is that a processor drawing 70 mA will use 70 mAH from the battery for every hour that it runs. Of course, a data logger is specifically designed to limit the amount of time the processor runs to conserve power. When the interval timer triggers the processor, it comes up and surveys the situation very quickly. It looks at nonvolatile **memory** for details about the present activity that may have been left from a previous reading. After this initialization, it reads the ADC channels and logic levels from the parallel I/O and either processes the data in some way or directly stores the values and increments the pointers. At the conclusion of these tasks, its last instruction is `PORT1=0`. This shuts off the processor.

The most important thing to note here is that the 80C52 is a very fast processor and all these activities happen very quickly. Such a program running entirely in BASIC only takes about 250 ms. I would hazard a guess that an assem-

LEARN MULTITASKING

Smx

COLLEGE KIT $95

The *smx* College Kit allows you to learn and to experiment with multitasking. It includes demo source code, a tutorial User's **Guide, a** Reference Manual, **and help files -- everything you need to try out multitasking! The CK is a reduced version of our** *simple* ***multitasking executive.*** **It runs at full speed and is ROM'able. Use it with MS-DOS or stand-alone. Works with the 80x88 family and** Microsoft or Turbo C **and assembler. The full version of** *smx* **is available for $1495 with no royalties.**

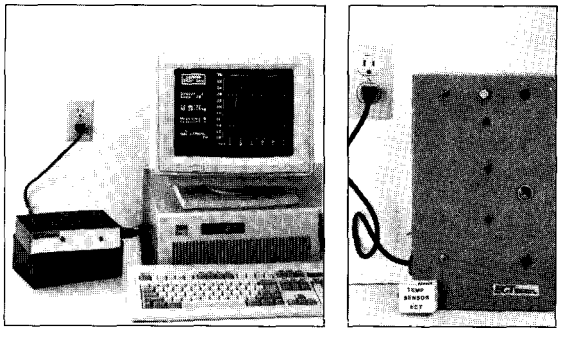
MICRO pd DIGITAL

1-800-366-2491

CYPRESS, CA 90630-5630

Reader Service #185

PC REMOTE I/O SYSTEM



Control, monitor home, office with complete remote data control system without special wiring to remote locations. System uses existing power lines for two-way communication between PC and remote modules. Remote has 4 analog in, 4 digital in and out. Analog output and sensors available.

$495 Basic Kit, Master and Remote Module, 4 temperature sensors. BASIC program and driver. Send for brochure.

ECT, INC.

4600 N.W. 9th Ct., Ste. 7, Plantation, FL 33317

305-587-3155

Reader Service #180

bly language version might take less than 50 ms (and Ed claims he can do it in under 2 ms).

Power is only consumed while the processor is running. During 250 ms at 70 mA, the processor used:

$$\frac{(250 \times 10^{-3}\,s)(70 \times 10\text{-}3\ A)}{(3600\ s/hr.)} = 4.8\ \mu AH$$

Each reading the data logger takes, even though it uses 70 mA, only consumes 4.8 µAH from the battery. The NiCd batteries used on my **prototype** are rated at 0.5 AH. At this consumption rate per reading, I could take over 100,000 readings; more than enough to fill the available memory and even ignore data during many of the samples.

Translating this to service life depends upon the interval. If we take five bytes per reading and have 30K bytes available, then there will be 6000 readings. If these readings are taken every three hours then we'll need the 0.5-AH battery on the interval timer because it will take two years. If a reading were taken every two hours (the limit of the fixed-interval unit), 6000 readings would still take 1.37 years!

At some point rational thought has to prevail and the shelf life of the batteries have to be considered. For extremely long duration monitoring, like the storage and shipment of vaccines, we would use lithium batteries in both the timer and the processor. New lithium batteries intended for use in cameras will supply the higher current the processor needs yet still have a IO-year shelf life. Memory management is a function of intelligence.

Temperatures inside shipping crates don't change that rapidly. Even if the data logger turns on every hour to sample the temperature it still only takes 210 mAH to run the data logger's processor section for five years!

For applications which take six months to two years, I suggest using alkaline batteries for the processor. Under six months, NiCd batteries are fine. They shouldn't be used for longer periods of time because they exhibit self discharge, which in this application, probably exceeds the logger's consumption. The NiCd batteries are fine for the UPS function, however.

SOFTWARE

I've already alluded to the software requirements of the Circuit Cellar Data Logger. It is really a trivial program in BASIC and I've already outlined how to reserve nonvolatile memory for your readings. Data logging is fairly simple: wake up, look around, take a reading, keep it or throw it away, turn off.

BACK TO THE SOLARIUM

So what did I determine after logging data in the solarium for a month? I built the solarium too well and it doesn't need any controls! How am I supposed to contribute to a home automation issue if the only thing I prove is that my home doesn't need any? I guess that's life.

The solarium seems to retain heat very well. Of course, that might have something to do with the fact that there is 18" of insulation in the roof and both the concrete floor and foundation have 2" of polystyrene insulation. The Pella windows used in the greenhouse are so efficient I'm tempted to replace all the rest of the glass in the house.

Even with all this glass and tile, the temperature remains fairly constant. During the middle of January, the solarium would reach the mid-80s from just sunlight if it were not vented. I installed a simple thermostat (here we go again) on a pair of variable-speed fans located near the ceiling that could exhaust the excess heat into the center of the house. With the thermostat set at 72°, the solarium stays at pleasant temperatures, even in bright sunlight.

During the evening, the wood stove is on and it seems to provide more than enough. Even with it running full blast, the room rarely exceeds 74-75 degrees. The exhaust fan comes on again to keep this temperature in check. When I retire for the evening, I will load up the stove once more. Even though the temperature outside is about 10°F, I can expect the solarium to still be at about 68-69 degrees in the morning.

This happens for a couple of reasons. As soon as the fire drops a bit, the exhaust fans will turn off and the heat from the firebrick-lined stove will stay in the room. This will maintain the temperature for 4-6 hours anyway. As the stove temperature drops, all the heat stored in the warm tiled concrete floor (calculated to be about 75,000 BTUs) continues to warm the room. The end result is that as long as I am willing to use a lot of wood in the stove, the environment is pleasant. The total control apparently needed is a thermostatically controlled fan.

It's always possible that summer will change these conclusions but before I install a 50-ton air conditioning system, I think I'll whip out my data logger and plot a few trends. Perhaps I won't have to write such a big check.

HARDLY A CONCLUSION

I completely dismissed my only opportunity to follow through on the theme of this issue. It appears that adding more attached structures will have the same results. The only opportunity I might have for demonstrating a useful example of closed-loop environmental control is by using something that most certainly would need it.

Rather than give up, I ordered an 11' x 20' greenhouse! Instead of the sun room protruding from the side of the solarium, this will be a working greenhouse with automatic watering, climate and humidity controls, heaters, automatic vents, and so on. If I can't make a case for computer controlling this mess efficiently then I better hang it up. Stay tuned.. .

Steve Ciarcia (pronounced "see-ARE-see-ah") is an electronics engineeyand computerconsultant with experience in process control, digital design and product development.

IRS

293 Very Useful
294 Moderately Useful
295 Not Useful

FEATURE ARTICLE

Build A Power Frequency Monitor

Ed Nisley

Counting Cycles Until It Hertz

Everyone knows that the standard United States AC power frequency is 60 Hertz. But, as with most things everyone knows, that isn't quite right. Right now the AC line in my office is running at 59.98 Hz. By late afternoon the line frequency will be down to 59.96 Hz, but after midnight it will rev up to 60.04 Hz for most of the early morning hours, before slowing to 59.94 around the start of first shift.

Although you may think those small variations are unimportant, they are critical to the digital alarm clock by your bed because it tells time by counting power line cycles. After 24 hours at 59.94 Hz, your clock will be 86 seconds slow; at the end of the month you'll be 45 minutes late for work. Because your digital clock is always spot on, you must be getting 5,184,000 cycles each and every day.

If you are interested in keeping track of your power line, you might want to build the five-chip power line frequency monitor described in this article. All of the frequency sampling and display control logic functions are handled by, yes, an 8031 processor, so three of those five chips are the requisite CPU, EPROM, and address latch. A 5x8 LED array shows a histogram of the instantaneous frequency samples.

The far left column of the display sketched in Figure 1 indicates the frequency resolution, which can range from 0.02 Hz to 0.50 Hz per column. The remaining seven columns display the number of frequency samples in each of seven bins centered around 60.00 Hz. The figure shows an AC line frequency between 59.98 and 60.00 Hz, with a power line glitch producing a higher frequency sample.

Those of you with 50-Hz power lines take note: the firmware automatically compensates for the difference. As you read the article, substitute "50 Hz" for "60 Hz" and you'll be on the right track.

POWER CYCLING

Most small projects get their juice from a "wall wart!' power supply that converts 120 volts AC into 5 volts DC. These supplies move the mysteries of power supply design into a (literally) black box so you can concentrate on the digital end of the project. Because we want to monitor the power line frequency we need access to the AC line voltage rather than a rectified and filtered DC supply.

Steve has covered power supply design in the past, so you should recognize the key parts in Figure 2 with no trouble. The transformer can have nearly any secondary voltage between 8 and 10 VAC RMS, although higher voltages increase the power dissipated in the 7805 regulator. The 1000-µF filter capacitor may be larger than you expect, but the circuitry draws over half an ampere.

The optoisolator produces a 60-Hz square wave at about 50% duty cycle from the transformer secondary

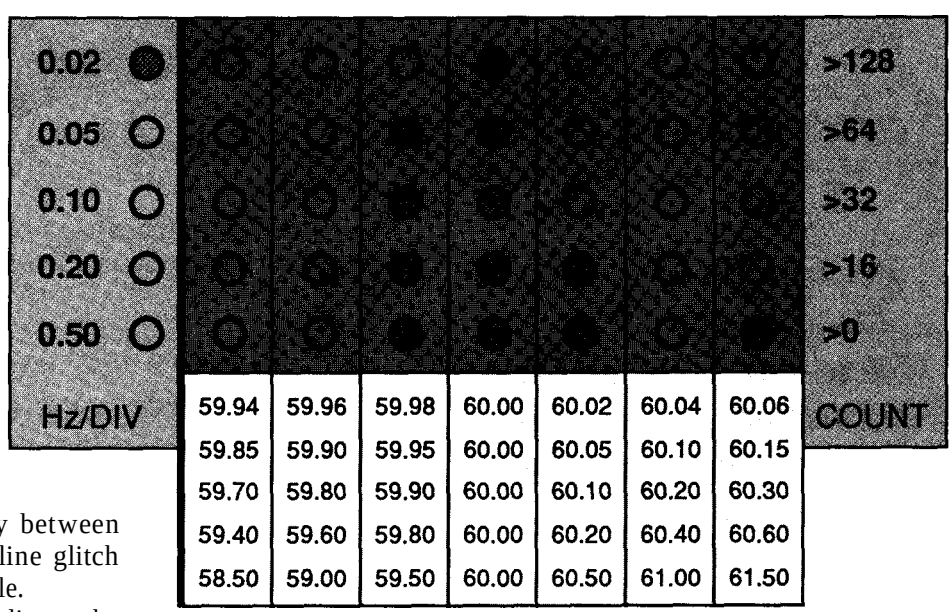


Figure 1 -The *fur left* column *of the display* indicates the *frequency resolution,* which can range from 0.02 Hz to 0.50 Hz *per* column. *The* remaining *seven columns display the number of frequency samples in each of seven* bins centered around *60.00* Hz. *The* figure shows an AC line frequency *between 59.98 and 60.W Hz, with a power line glitch producing a higher frequency* sample.

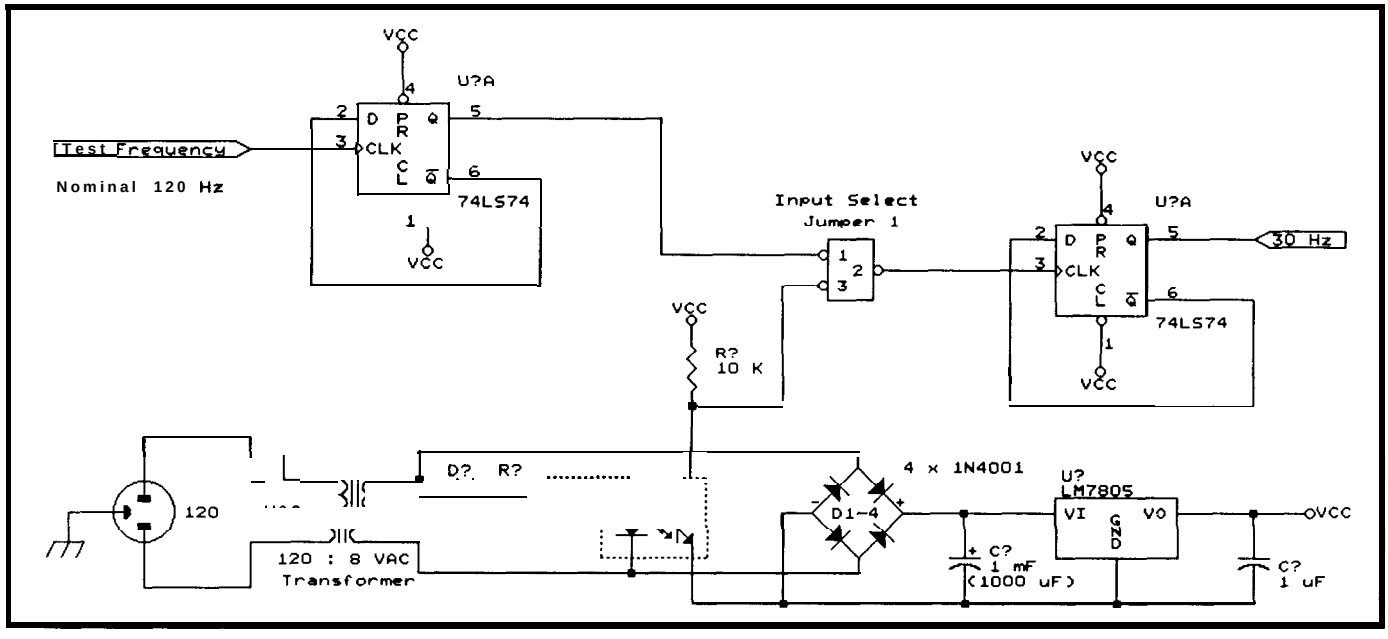


Figure *P-Anyone familiar with* **power** *supply design* ***will have no trouble understanding the components of the monitor. The transformer can have nearly any secondary voltage between 8 and 10 VAC RMS. The 1000-microfarad filter capacitor may be larger than you expect, but the circuitry draws over half an ampere. The optoisolator produces a 60-Hz square wave at about 50% duty cycle from the transformer secondary voltage. The 330-ohm resistor in series with the LED limits the peak current to about 30*** *mA.*

voltage. The 330-ohm resistor in series with the LED limits the peak current to about 30 mA; the value and power rating depend on the transformer secondary voltage. Although it is tempting to use the optoisolator LED as one of the bridge rectifier diodes, its forward current and reverse voltage ratings are both far too small.

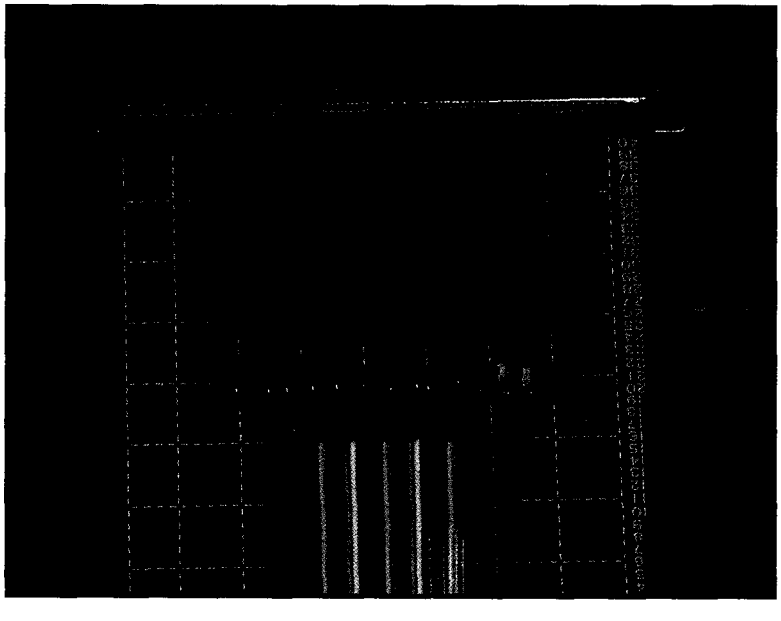
Because it is not practical to have the utility company vary the line frequency while you check out your circuitry, Jumper 1 selects either the power line frequency or an external test signal from a function generator. The 74LS74 flip-flops divide their input frequency in half, so the output signal is either 30 Hz or one quarter of the test frequency (nominally 120 Hz). In either case, the final output has exactly 50% duty cycle: it is ON for one full 60-Hz cycle and OFF for the next.

There are many power supply variations possible depending on your skill and available hardware. Any design is OK as long as it provides both 5 volts DC at about 750 mA and a 30-Hz square wave derived from the power line.

MARKING TIME

There are basically two ways to measure frequency: count the number of cycles in a given time interval, or measure the duration of a single cycle. The former gives you the frequency directly in cycles/second, while the latter produces the period in seconds/cycle. Which method you use depends on both your goals and the available hardware.

Because we are interested in very small frequency variations, the first method requires a long time interval. For example, it takes 50 seconds to distinguish between 60.02 Hz and 60.00 Hz because that's the shortest interval at which they differ by a full cycle. There are tricky ways to reduce the time, but updating the display every minute or so is not really acceptable.



On the other hand, timing a single cycle gives you an instantaneous period value that may reflect moment-to-moment jitter rather than a true frequency variation. Accumulating several samples can reduce this variation, but you need to be careful not to average out the significant deviations. In this case, I simply display all the counts in a histogram format and let your eyes do the averaging.

The 8031 hardware can handle either

2x4"s™

Complete 2"x4"Board Line...

NMIS-0021 F68HC11 CPU $99
NMIS-0016 SAB80535 CPU $99
NMIS-0001 PROTOTYPE CARD $19
NMIS-1022 65C22 VIA $49
NMIS-1055 82C55 PPI $49
NMIS-3002 32 BIT HC I/O $80
NMIS-3003 64 BIT HC INPUT $80
NMIS-3004 64 BIT HC OUTPUT $80
NMIS-4000 2CH 8 BIT D/A (up to 6 chs can be added) $65
NMIS-4002 4CH 12 BIT D/A (up to 7 chs can be added) $99
NMIS-4004 8CH 12 BIT MADC3742 A/D $99
NMIS-5000 SER I/F ACIA RS-232, 422, 485 ..... $65
NMIS-5002 SER I/F DUAL ACIA RS-232 ..... $95
NMIS-7000 16 BIT OPTOISOLATOR INPUT $80
NMIS-7001 16 BIT OPTOISOLATOR OUTPUT $80
NMIS-7011 8 400V 1.5A SOLID STATE RELAYS $99
NMIS-7021 8 SPDT 125V 2A MECH RELAYS $95
NMIS-7040 4CH LOW POWER STEPPER MOTOR CONTROLLER $95
NMIS-9002 3CH 16 BIT COUNTER $45
NMIS-9003 REAL TIME CLOCK $45

...none over $100 !

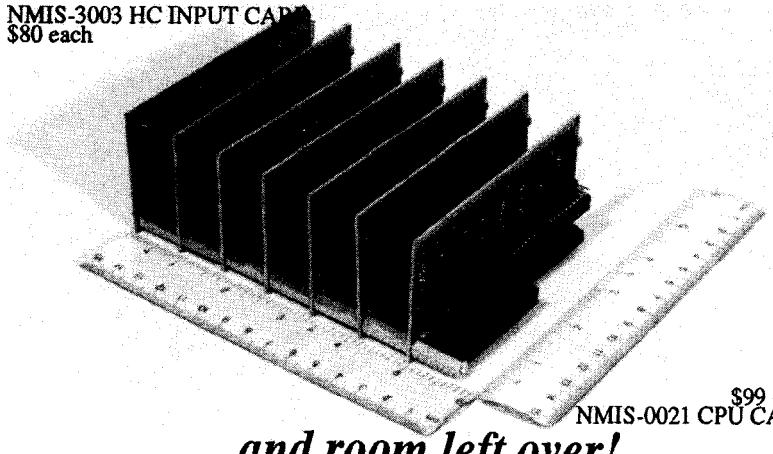
New Micros Inc. 1601 Chalk Hill Rd., Dallas, Tx 75212 (214)339-2204

2"x4"x6"

by 384 Discrete Inputs...

...plus up to 64K of Battery/Super Cap Backed RAM and F68HC11 CPU with Operating System and FORTH language in builtin ROM, 8 Chs of 8 Bit A/D, Watchdog Timer, Major Counter/Timer Subsystem, 1/2K EEPROM, 1 Async, 1 Sync Serial Port, 3 Edge Sensitive Inputs, 8 Bit Pulse Accumulator..,

NMIS-3003 HC INPUT CARD
$80 each



$99 each
NMIS-0021 CPU CARD

...and room left over!

New Micros Inc. 1601 Chalk Hill Rd., Dallas, Tx 75212 (214)339-2204

Reader Service # 188

method, but the latter is particularly easy because the on-chip timers have external gate inputs. Figure 3 shows how simple the circuit can be. There are only two inputs: the 30-Hz timing signal and a push button to select the display range. Apart from the LED scanning signals, there is only one output: a scope sync pulse marking the start of Anode A's activity to simplify debugging the driver circuitry.

The 30-Hz signal actually performs two functions within the 8031: Timer 0 runs only while pin 12 is high, and INTO goes active on when pin 12 goes from high to low. Because the 8031 runs at 12 MHz, the value in Timer 0 is precisely the period of the previous power line cycle in microseconds, and there is even an interrupt available to tell us when the count is valid!

Converting period into frequency requires a division, and an extended-precision one at that. On an 8031, this can be a nightmare because the CPU can divide one unsigned byte into another.. .anything more complex being a matter of firmware. After I describe the display hardware I'll explain how I completely sidestepped this problem.

SCANNING REDUX

Jeff described how to scan a dot-matrix LED array in "From the Bench' in issues 13 and 14 of CIRCUIT CELLAR INK, so I need not go into a lot of detail. Figure 4 illustrates the frequency monitor's LED array and driver circuitry. While Jeff and I use the same LED array (I swiped one right off his heap), you can wire up 40 individual LEDs to get much the same effect. Also, the discrete transistors are there just to show that fancy integrated circuits aren't the only way to go.

Incidentally, if you use discrete LEDs, you could make the center three columns green, the next pair yellow, and the outer columns red. Talk about an eye-catcher! ***[Editor's Note:*** *LED modules similar to that used by Ed are available with tricolor LEDs, so if's possible to build a three-color display with LED modules.1*

As you saw in Figure 1, the array is "on its side" relative to Jeff's scrolling LED display. The array doesn't carewhichwayitisoriented,but what were once rows are now columns (and vice versa), so you must be careful with terminology. I refer to "anodes" and "cathodes" to reduce the confusion in this article, but you can think "columns" and "rows" if you like. Remember that the cathode is the negative end of the LED (the end with the bar) and you'll have no trouble.

The 8031 displays data by writing the bits to Port 1, which activates the eight 2N2907A transistors driving the LED anodes. It then turns on one of the BS170 FETs to draw current through one of the rows. The only LEDs that light are theones connected to both an active anode driver and the active cathode driver, so the others stay off.

Because the BS170 FETs are rated for only 500-mA drain current, the 39-ohmresistorslimit thecurrent through each LED to about 60 mA. The resistor value is calculated from:

$$\frac{V_{supply} - V_{CE(sat)} - V_{LED} - V_{DS}}{I_{LED}}$$

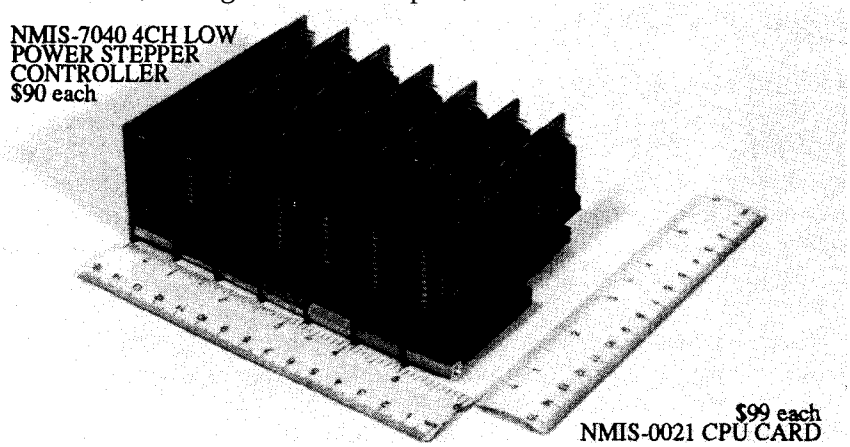
In our case, this works out to be:

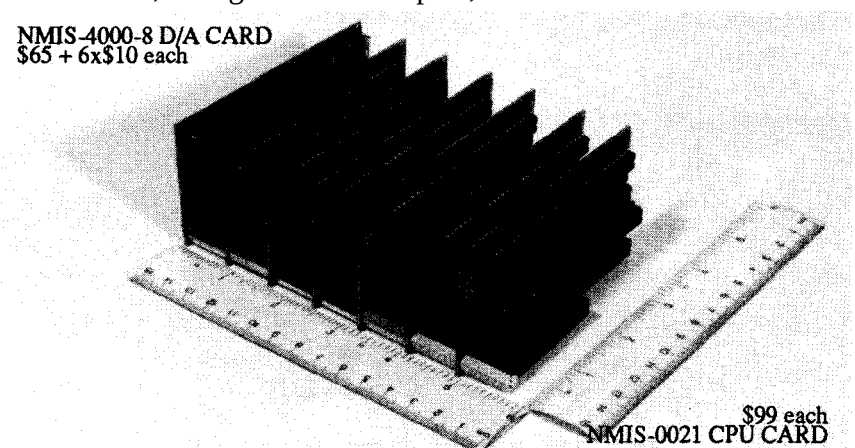
$$\frac{5 - 0.5 - 0.75 - 1.5}{0.060} = 37.5 \text{ ohms}$$

You must calculate the FET's V_{DS} value from the r_{DS} value times the total 500-mA cathode current rather than the 60-mA anode current for each LED. Obviously, this voltage varies depending on the number of LEDs turned on, so you need to think about the limiting cases.

Interrupts from Timer 1 pace the scanning, with each cathode driver active for 1667 microseconds. There are thus about two complete scans per power line cycle, but the intervals are not locked together. The cathode driver data comes from five 8031 Internal RAM bytes, so the display is truly bit-mapped!

Indeed, the power-on routine scrolls test patterns around the display by directly manipulating the Internal RAM bytes. All bit-map RAM

2"x4"x6"
by 24 Stepper Motor Ports...
...plus up to 64K of Battery/Super Cap Backed RAM and F68HC11 CPU with Operating System and FORTH language in builtin ROM, 8 Chs of 8 Bit A/D, Watchdog Timer, Major Counter/Timer Subsystem, 1/2K EEPROM, 1 Async, 1 Sync Serial Port, 3 Edge Sensitive Inputs, 8 Bit Pulse Accumulator...
NMIS-7040 4CH LOW POWER STEPPER CONTROLLER $90 each

$99 each
NMIS-0021 CPU CARD
...and room left over!
New Micros Inc. 1601 Chalk Hill Rd., Dallas, Tx 75212 (214)339-2204

2"x4"x6"
by 48 D/A Outputs...
...plus up to 64K of Battery/Super Cap Backed RAM and F68HC11 CPU with Operating System and FORTH language in builtin ROM, 8 Chs of 8 Bit A/D, Watchdog Timer, Major Counter/Timer Subsystem, 1/2K EEPROM, 1 Async, 1 Sync Serial Port, 3 Edge Sensitive Inputs, 8 Bit Pulse Accumulator...
NMIS-4000-8 D/A CARD $65 + 6x$10 each

$99 each
NMIS-0021 CPU CARD
...and room left over!
New Micros Inc. 1601 Chalk Hill Rd., Dallas, Tx 75212 (214)339-2204
Reader Service #188

changes must be synchronized with the display scan to prevent flicker, but this is easily handled with a single bit set in the Timer 1 interrupt handler.

That interrupt routine also scans the button to decide when to change ranges. It debounces the signal by requiring two consecutive ON samples, then increments a counter that records how long the button was pressed. I won't go into the details of this, but the source code on the BBS uses an interesting trick to simplify the sequential logic. [Editor's **Note:** *Soffwayefoy fhisayficleis available for downloading from* the Circuit Cellar BBS, *or* on Software *On Disk* #15. *For downloading and ordering information, see page 77.1*

PAINLESS RECIPROCALS

Every other power line cycle fills Timer 0 with the period in microseconds. The firmware must take the reciprocal of that number to get the frequency, which seems to imply a division of some sort. Fortunately, there aren't that many possible periods.. .so the firmware looks the answer up in a table!

The button selects one of five display ranges, from0.02 Hz per column to 0.50 Hz/column. With seven columns in the display, the widest range handles frequencies between 58.50 Hz and 61.50 Hz, corresponding to periods between 17,094 and 16,260 µs. The timer resolution is one microsecond, so there are only about 800 possible periods; that's small enough to make table look-up practical.

Because the frequencies are quite near 60 Hz, it makes sense to put (scale factor)*(actual frequency – 60.00 Hz) in the table, rather than the value in Hertz. Listing 1 shows the assembly source for sections of the 60-Hz table. There is a similar table for values near 50 Hz. The tables actually have about 1200 entries each so you don't have to regenerate them if you tinker with the display ranges.

Of course, I didn't generate those tables by hand! Instead, I wrote a REXX program (you could use C, Pascal, BASIC, or whatever is popular this week) to create assembler source code. The program is available on the BBS so you have a starting point for creating your own tables (if you live where the AC line frequency is, say, 400 Hz).

Listing 2 shows what's involved in converting the period to a frequency. Because the table is in EPROM, the firmware must use MOVC rather than MOVX to fetch the byte. Much easier than a longdivision routine, right?

DOTS FROM COUNTS

The remainder of the program is straightforward. The code converts the frequency value to an index that selects

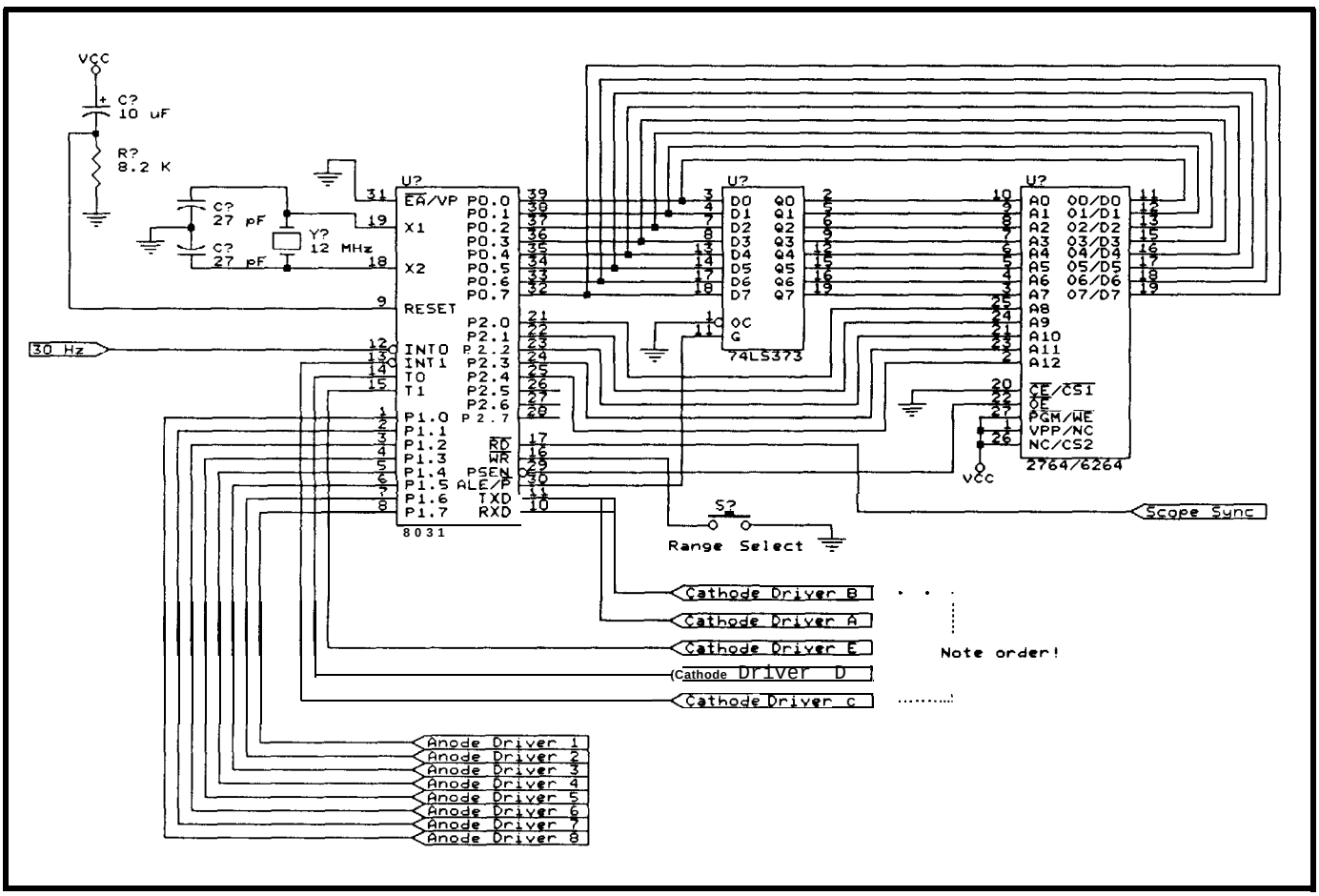


Figure 3—*An803 7 can make* **for a very simple circuit. There are only two inputs: the 304-k timing signal and a push** *button to* **select the** *display range. Apart from the LED scanning signals, there is only one output: a scope sync pulse marking the start of Anode A's activity.*

one of seven histogram bins; each sample thus increments one bin, with the bin values limited to a maximum of FF hex. After each increment, the code converts all seven counts into five-bit "thermometer" bar codes. All of those conversions are done through tables, so there is very little calculation going on by the micro and the process is quite rapid.

One minor complexity arises because the anode drivers "light up" a horizontal row, but the thermometer codes corresponding to each bin lie along vertical columns. A scan conversion routine transposes rows and columns to get the dots in the right places.

A separate routine turns on a single bit in the left column to indicate which frequency range is currently active. When you build this gizmo, you might want to use a 5x7 LED array and wire five discrete LEDs off to the side to indicate the display range; cramming it into a single LED brick like I did makes the display somewhat confusing at times.

Finally, the five bytes resulting from all those tables are copied into the anode driver buffer just after the next Timer 1 interrupt. The interrupt routine updates the display 30 times a second, so it shows **truly real-time** information!

CALIBRATION AND VERIFICATION

because the **8031** timers use the CPU oscillator to time the incoming periods, the ultimate accuracy of the frequency monitor depends on the crystal's stability. The data sheets say that the crystal is within 0.005% of nominal, which works out to 0.003 Hz around 60.00 Hz. The smallest display increment is 0.02 Hz, which is 0.033%, so the oscillator is certainly accurate enough for our purposes.

Long-term drift and temperature stability are other issues, but typical (read "cheap") oscillators seem to run around 0.01% for moderate changes in temperatures and times, so that's close enough, too, because this instrument will be used in a shirt-sleeve environment.

My Fluke meter (accurate to 0.05%) tracks quite nicely with the display, which is comforting, but not conclusive. My ferroresonant UPS has a diagnostic microprocessor that **reports a** frequency within 0.01 Hz of the LED display. In short, the frequency monitor passes the dipstick test.

so...

All of your home control projects depend on AC power, so isn't it time to get a handle on what's coming out

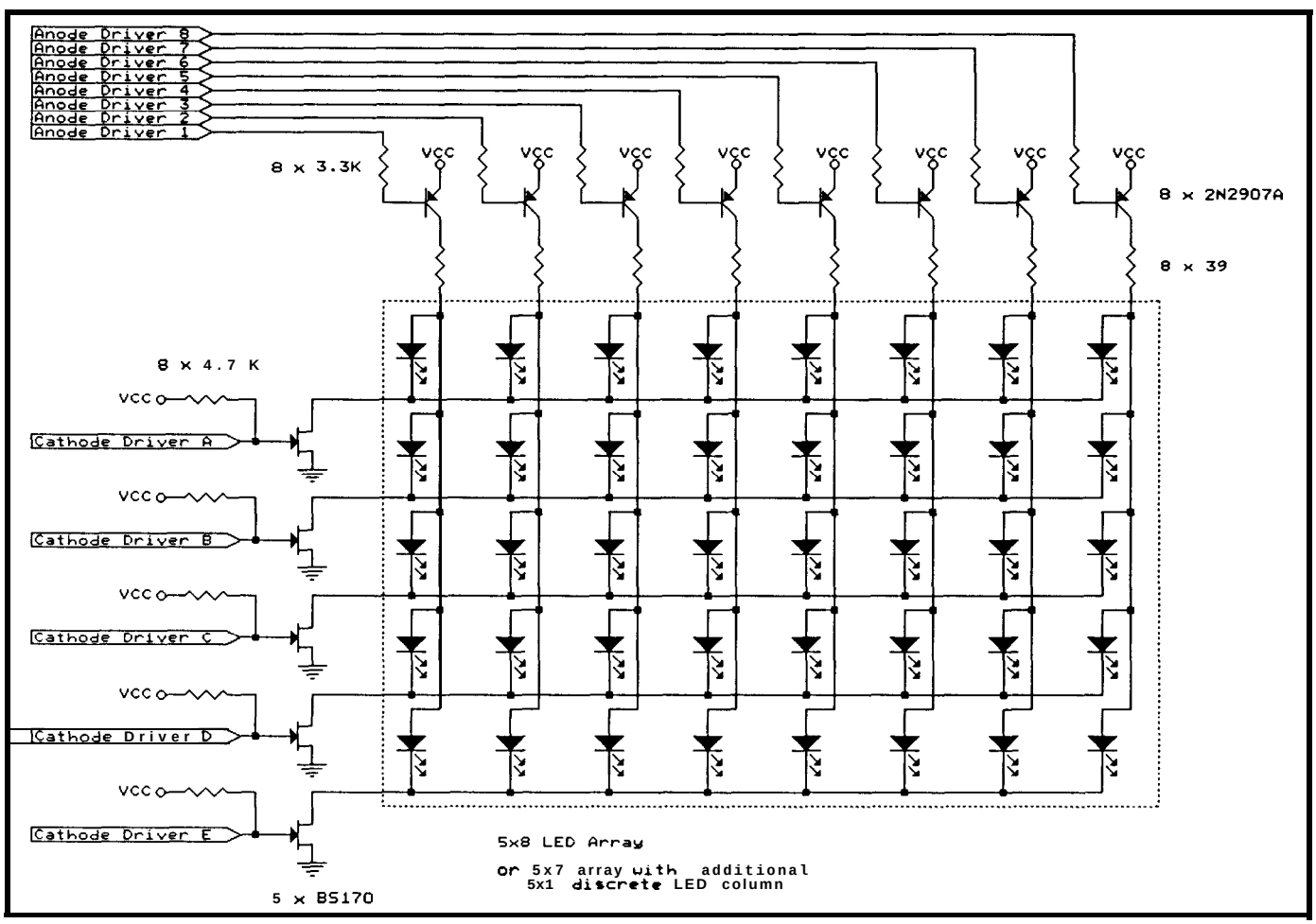


Figure 4—*The* frequency ***monitor's*** LED ***array and driver circuitry. You can wire up 40 individual LEDs to get much the same effect. The discrete transistors are there just to show that fancy integrated circuits aren't the only way to go.***

```
; FREQ60.A51
; Center freq : 60 Hz
; Table size : 0480 hex
; Scale factor: 60 dec
MinPd60         DW      3f00h
                PUBLIC  MinPd60

MaxPd60         DW      437fh
                PUBLIC  MaxPd60

Freq60Table EQU     $
                PUBLIC  Freq60Table

     DB 120  ; 78h 62.003 Hz, 16.128 ms, 3F00h
     DB 120  ; 78h 62.000 Hz, 16.129 ms, 3F01h
     DB 120  ; 78h 61.996 Hz, 16.130 ms, 3F02h
     DB 120  ; 78h 61.992 Hz, 16.131 ms, 3F03h
     DB 119  ; 77h 61.988 Hz, 16.132 ms, 3F04h

<<<< lines omitted >>>>

     DB 2    ; 02h 60.031 Hz,  16.658 ms,  4112h
     DB 2    ; 02h 60.027 Hz,  16.659 ms,  4113h
     DB 1    ; 01h 60.024 Hz,  16.660 ms,  4114h
     DB 1    ; 01h 60.020 Hz.  16.661 ms,  4115h
     DB 1    ; 01h 60.016 Hz:  16.662 ms,  4116h
     DB 1    ; 01h 60.013 Hz,  16.663 ms,  4117h
     DB 1    ; 01h 60.009 Hz,  16.664 ms,  4118h
     DB 0    ; 00h 60.006 Hz,  16.665 ms,  4119h
     DB  0   ; 00h 60.002 Hz,  16.666 ms,  411Ah
     DB  -0  ; 00h 59.998 Hz,  16.667 ms,  411Bh
     DB  -0  ; 00h 59.995 Hz.  16.668 ms,  411Ch
     DB  -1  ; FFh 59.991 Hz,  16.669 ms,  411Dh
     DB -1   ; FFh 59.988 Hz,  16.670 ms,  411Eh
     DB -1   ; FFh 59.984 Hz,  16.671 ms,  411Fh
     DB -1   ; FFh 59.980 Hz,  16.672 ms,  4120h
     DB -1   ; FFh 59.917 Hz,  16.673 ms,  4121h
     DB -2   ; FEh 59.973 Hz,  16.674 ms,  4122h
     DB  -2  ; FEh 59.970 Hz,  16.675 ms,  4123h

<<<< lines omitted >>>>

     DB  -127; 81h 57.883 Hz, 17.276 ms, 437Ch
     DB  -127; 81h 57.880 Hz, 17.277 ms, 437Dh
     DB -127; 81h 57.877 Hz, 17.278 ms, 437Eh
     DB -128; 80h 57.873 Hz, 17.279 ms, 437Fh

     END
```

listing 1 -*Conversion table from periods in microseconds to frequencies in Hertz. The* table index is:
(table base address) + ((period in microseconds) - 3f00)
***The* table entries are: 60 . *(frequency - 60.00)* expressed in twos complement notation.**

of the outlet? Besides, if you do a nice job packaging the monitor, it makes a great conversation starter.

You could also add a function to accumulate the minimum, maximum, and average frequencies for each hour and report them over the serial port. You must rearrange the cathode driver outputs to free up the serial pin, but then you can record the values on a PC and do some trend analysis. Hey, you could find the Fourier transform of the line frequency variation.. .

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

IRS

296 Very Useful
297 Moderately Useful
298 Not Useful

```
;-------
; Convert period into frequency value

GetFrequency PROC
        MOV     A,PeriodLo      ; subtract
        CLR     C               ;  lowest
        SUBB    A,MinPeriod+1   ;   period
        MOV     DPL,A
        MOV     A,PeriodHi
        SUBB    A,MinPeriod
        MOV     DPH,A
        JNC     L?lowOK         ; if too small,
        MOV     DPH,TableBase   ;  use lowest
        MOV     DPL,TableBase+1;   period
        SJMP    L?getfreq

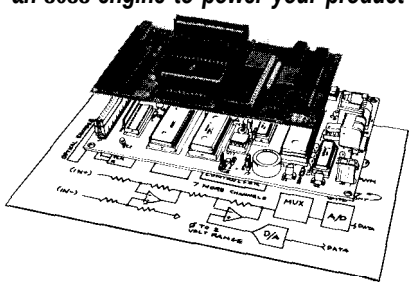
L?lowOK
        MOV     A,DPL
        CLR     C
        SUBB    A,TableSize+1
        MOV     A,DPH
        SUBB    A,TableSize
        JC      L?highOK        ; if too big,
        MOV     DPH,TableSize   ;  use highest
        MOV     DPL,TableSize+1;   period
        DecDPTR

L?highOK
        MOV     A,TableBase+1   ; tack on
        ADD     A,DPL           ;  table base
        MOV     DPL,A           ;   address
        MOV     A,TableBase
        ADDC    A,DPH
        MOV     DPH,A

L?getfreq
        CLR     A
        MOVC    A,@A+DPTR       ; fetch the byte...
        RET
GetFrequency ENDPROC
```

listing 2—*The current period* is at *PeriodLo* and *PeriodLo+ 1* in the 8031's *internal* RAM. This routine converts that *16-bit* value into an *index into the table shown in Listing 1* and returns the corresponding *table entry* in the accumulator.

The MICRO88
an 8088 engine to power your product



This small (3" x 5"), pre-engineered, μ module is designed to mount to your application Specific interface board for minimum cost, no-snag development. Onboard resources include: watchdog timer, dual UART with RS-232 and RS-485, 8088 μ, 64K ROM space and 32K RAM space. Fully integrated high level language development environment (Basic, C, Forth). We specialize in custom applications, both hardware and software.

VESTA TECHNOLOGY
7100 W. 44th Ave., Suite 101, **Wheat Ridge,** CO 80033
(303) **422-8088,** FAX 422-9800, **BBS** 278-0364

Reader Service #191

Building étude

Part 2

A 25-MHz Analog-to-Digital Converter for the PC Bus

J. Conrad Hubert
Dick Hubert

In the first half of this project (see "Building étude: Part 1," CIRCUIT CELLAR INK #13), we showed the hardware side to a 25-MHz analog-to-digital converter for the PC bus. With the hardware in place, we can switch our attention to the software that allows you to build applications around the hardware.

Programming is a lot like religion-everyone has their own personal beliefs. We profess that software which pretends to be all things to all users is unnecessarily cumbersome and still may not provide the features needed for your specific project. In light of that, we've elected to provide the source code to etude's driver routines. This allows you to create a manageable software package for your unique application. The programs presented here are written in Turbo Pascal because it's as understandable as pseudo code, yet will actually compile. We also think you'll find Turbo Pascal a straightforward translation into your favorite programming language. ***[Editor's Note:*** *Software for Cellar BBS, or on Software On Disk #15. See page 77 for downloading and ordering information.]*

```
program Offset; { Allows you to observe the effect of adjusting }
                { étude's offset potentiometer (R3)              }

uses       { This makes the driver available to the application }
  ADCDRV;

const
  Base Address : word = $250;                        { 01 0101 0000 }
  MemoryModel : longint = 4 * 1024  { 4096 in Turbo version 4.0 }

BEGIN                                                { Ctrl-break to exit }
  _AssignRegisters(BaseAddress);{ MUST be FIRST routine called }
  _InitializeSmartChips (MemoryModel);    { NEXT routine called }
  repeat
    writeln (_ReadADC);
  until false
END.                                                          { main }
```

listing 1 *-A sample application program serves to showhoweasyitis to use étude's driver* **routines.**

Before moving on, there's one naming convention you should know about. An underscore, as a prefix a procedure name, indicates that excerpts from the driver PAS as shown in Listing 2. Without further ado... the nitty-gritty.

The scope of ADCDRV's interface section is global. Constants, types, variables, functions, and procedures defined in the interface section are visible outside the unit. Conversely, the scope of the driver's implementation section is local, and definitions are not visible outside the unit.

BASIC DRIVER FUNCTIONS

_CacheReadWrite affords control over the cache's RD\ (PC6) and WR\ (PC5) lines. (Memory chips in the 4K-byte version of étude do not have a RD\ line, and therefore not writing implies a read.) Whenever the ADC is emanating data, the cache must not be set to write, or outputs will be in contention for the cache bus. In a similar manner, if you reprogram the 8255's port A for output to the cache, the ADC's output enable line must be high and the cache must be set for read.

_ADC_OutputBuffer controls the ADC's output enable (OE\) line via 8255 port C7. When PC7 is high, the ADC's output buffer is in the high impedance state.

_MUX_ChannelSelect controls the 1-of-8 multiplexer (74LS151) input select lines (S0, S1, S2) from 8255 ports B7, B6, and B5, respectively. The mux outputs are only active when the cache is not full. A full cache is defined as all cache address bits set (i.e., the cache pointer is at the last storage element).

_TTL_Buffer enables/disables one-half of the I/O buffer (74LS244) via 8255 port C0. It is used to simultaneously disallow the DB15 hardware inputs Inhibit-Release\ (pin 8) and External Timebase (pin 6). It also tristates the hardware output Done\ (pin 2).

_AddressGenerator permits inhibiting the address generator from software via 8255 port C4. When the aforementioned line is high and pin 1 of the DB15 (Inhibit-Release\) is not pulled low, the address generator increments once per sample. If either line goes low, the address generator and sample counter are inhibited but the ADC still makes a conversion. This has several uses:

1) Since the ADC has a two-conversion pipeline, you may avoid getting "stale" data from the converter.

2) Data acquisition may be synchronized via software and/or hardware.

3) It permits data to be read in real time from the converter without disturbing the contents of the cache.

ACCESSING ADDRESSES

_DirectMemoryAccess permits asserting the PC's DACK\ (DMA Acknowledge) and DRQ (DMA Request) lines. When PB2 is high, the DACK\ line tristates DACK\ and DRQ. When PB2 is low, DRQ goes high, requesting a DMA transfer, and the DACK\ line is allowed to pass through to the mux.

_AssignRegisters sets the global addresses of hardware registers relative to the étude's base address. Alterability of the base address allows up to 16 études in one computer. étude may reside at any of the I/O port locations shown in Figure 1. A read or write to I/O space is "in range" when the following conditions are met:

```
procedure _CacheReadWrite (Operation : DataFlow);
begin
  BitsC := BitsC and $9F;    { 1001 1111, mask read/write bits }
  case Operation of
    Read  : BitsC := BitsC or $20;        { x01x xxxx, set PC5 }
    Write : BitsC := BitsC or $40;        { x10x xxxx, set PC6 }
  end;
  port[ LatchC ] := BitsC;
end;

procedure _ADC_OutputBuffer (Operation : State);
begin
  case Operation of
    Enable  : BitsC := BitsC and $7F;  { 0xxx xxxx, clear PC7 }
    Disable : BitsC := BitsC or $80;   { 1xxx xxxx, set   PC7 }
  end;
  port[ LatchC ] := BitsC;
end;

procedure _MUX_ChannelSelect (Source : Channel);
begin
  BitsB := BitsB and $1F;{0001 1111 MUX bits PB7, PB6, PB5}
  case Source of
    OneShot         : begin end;                { I0 = 000x xxxx }
    MHz5            : BitsB := BitsB or $3F; { I1 = 001x xxxx }
    ExtInput        : BitsB := BitsB or $5F; { I2 = 010x xxxx }
    MHz20           : BitsB := BitsB or $7F; { I3 = 011x xxxx }
    VariableDivisor : BitsB := BitsB or $9F; { I4 = 100x xxxx }
    MHz10           : BitsB := BitsB or $BF; { I5 = 101x xxxx }
    IO_Read         : BitsB := BitsB or $DF; { I6 = 110x xxxx }
    MHz25           : BitsB := BitsB or $FF; { I7 = 111x xxxx }
  end;
  port[ LatchB ] := BitsB;
end;

procedure _TTL_Buffer (Operation : State);
begin
  case Operation of
    Enable  : BitsC := BitsC and $FE;  { xxxx xxx0, clear PC0 }
    Disable : BitsC := BitsC or $01;   { xxxx xxx1, set PC0   }
  end;
  port[ LatchC ] := BitsC;
end;

procedure _AddressGenerator (Operation : State);
begin
  case Operation of
    Enable  : BitsC := BitsC or $10;  { xxx1 xxxx, set  PC4  }
    Disable : BitsC := BitsC and $EF; { xxx0 xxxx, clear PC4 }
  end;
  port[ LatchC ] := BitsC;
end;

procedure _DirectMemoryAccess (Operation : State);
begin
  case Operation of
    Enable  : BitsB := BitsB and $FB;  { xxxx x0xx, clear PB2 }
    Disable : BitsB := BitsB or $04;   { xxxx x1xx, set  PB2  }
  end;
  port[ LatchB ] := BitsB;
end;

procedure _AssignRegisters (BaseAddress : word);
begin
  SampleCounter        := BaseAddress + 0;  { Counter/Timer 0   }
  SampleRateDivisor    := BaseAddress + 1;  { Counter/Timer 1   }
  IncAddressGenerator  := BaseAddress + 2;  { Counter/Timer 2   }
  Ctrl8253             := BaseAddress + 3;  { 8253 Control port}
  CacheBus             := BaseAddress + 4;  { Port A input      }
  LatchB               := BaseAddress + 5;  { Port B output     }
  LatchC               := BaseAddress + 6;  { Port C output     }
  Ctrl8255             := BaseAddress + 7;  { 8255 Control port}
end;

procedure _InitializeSmartChips (MemoryModel : longint);
var
  Coercion : real;
begin
  port[ Ctrl8253 ] := $30; { SampleCounter   mode 0 LSB/MSB     }
  port[ Ctrl8253 ] := $76; { SampleRateDivisor mode 3 LSB/MSB   }
  port[ Ctrl8253 ] := $98; { IncAddressCounter mode 4 LSB only }
```

(continued)

Listing 2—*Excerpts from étude's driver routines are more fully explained in the article.*

```
  port[ Ctrl8255 ]:= $90: { A = input, B = output, C = output }
  BitsB := $FF;           { 1111 1111.  set all bits high      }
  BitsC := $FF;           { 1111 1111,  set all bits high      }
  port[ LatchB ] := BitsB;
  port[ LatchC ] := BitsC;
  Coercion := (MemoryModel div 16) - 1;
  CacheFactor := trunc(Coercion);{ Cache factor is of type word)
end;

procedure _ZeroAddressGenerator;
begin
  port[ LatchC ] := BitsC and $FB; { xxxx x0xx Make PC2 low    }
  BitsC := BitsC or $04;           { xxxx xlxx Assure PC2 high }
  port[ LatchC ]:= BitsC;
end;

procedure _LoadSampleCounter (NumberOfSamples : longint);
var
  Noverl6 : word:
  Coercion : real;
begin
  Coercion := NumberOfSamples / 16;
  if Trunc (Coercion) < 2
    then Nover16 := 1
    else Nover16 := trunc(Coercion)-2;
  port1 SampleCounter ] := lo(Nover16); { LSByte first }
  port[ SampleCounter ] := hi(Nover16); { MSByte next }
end;

procedure _SetSampleRate (Frequency : real);
var
  Divisor : word:
begin
  if (Frequency <= 1.25e6) and (Frequency > 38)
    then begin
      Divisor := round (2.5e6 / Frequency);
      port[ SampleRateDivisor]:= lo (Divisor);
      port[ SampleRateDivisor ]:= hi (Divisor);
       MUX_ChannelSelect (VariableDivisor);
    end
  else if Frequency = 25.0e6
    then _MUX_ChannelSelect (MHz25)
  else if Frequency = 20.0e6
    then _MUX_ChannelSelect (MHz20)
  else if Frequency = 10.0e6
    then _MUX_ChannelSelect (MHz10)
  else if Frequency = 5.0e6
    then _MUX_ChannelSelect (MHz5)
  else if Frequency < 0 { Negative frequency ==> external osc. }
    then _MUX_ChannelSelect (ExtInput);
end;

function _SamplesRemaining: word:
var
  LSB, MSB : byte;
  Samples : word:
begin
  port[ Ctr18253 ] := $00; { 0000 XXXX Latch the SampleCounter }
  LSB := port[ SampleCounter ];          { LSByte read first }
  MSB := port[ SampleCounter ];          { MSByte read next }
  Samples := (MSB shl 8) + LSB;
  if Samples > CacheFactor
    then _SamplesRemaining := 0
    else _SamplesRemaining := Samples;
end;

procedure _SetOutputCoding (OutputFormat: Coding);
begin
  BitsB := BitsB and $E7; { 1110 0111, mask Output Coding bits }
  case OutputFormat of
    InvertedBinary   : beain end:                  { xxx0 0xxx. }
    TrueBinary       : BitsB := BitsB or $18;      { xxx1 lxxx }
    TrueTwosComp     : BitsB := BitsB or $08;      { xxx0 1xxx }
    InvertedTwosComp : BitsB := BitsB or $10;      { xxx1 oxxx }
  end:
  port[ LatchB ] := BitsB;
end:

procedure _FillCacheViaADC (Frequency : real;
                            NumberOfSamples : longint);
begin
  _CacheReadWrite (Write):
```

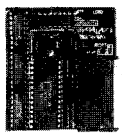
(continued)

Listing *2-continued*

DEVELOPMENT TOOLS

For the IBM PC and Compatibles

2764 - 27256 ROM EMULATOR



Appears as EPROM to target system • Connects to PC parallel port



- Plugs into target ROM socket and connects to PC parallel port via **modular telephone cable**
- Accepts 8K x 8 or 32K x 8 SRAM or NV SRAM
- **Reset output** restarts target after downloading
- Uses HC/HCT logic for CMOS & TTL compatibility
- Loads Intel Hex, Motorola S, hex, and binary files
- Command line software can be run from **batch files** for automatic downloading after assembly

$99 (without memory) **$119** (with 32K x 8 SRAM) **$139** (with 32K x 8 NV SRAM)

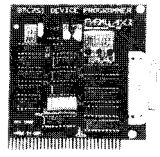
8051 FAMILY ASSEMBLER

- Works on all 8051 derivatives
- Supports standard Intel syntax
- Allows local labels and include files
- Labels may be up to 32 characters
- Generates-assembly listings
- Outputs Intel Hex

$59

Quick • Dependable • Clean Operation

87C751 DEVELOPMENT TOOLS



87C751 Programmer

Programs Signetics 87C751 and 87C752 microcontrollers. Handles EPROM, key, and security bits Loads Intel Hex, ASCII hex, and binary files

Programmer – DIP (includes DIP socket) **$289**

Programmer – PLCC (includes PLCC socket) **$229**



87C751 Bondout Board

"Mini-ICE" for the 87C751 Provides external EPROM socket for program memory Very useful when combined with ROM emulator

Bondout Board **$239**

Bondout Board and ROM Emulator **$338**

PARALLAX INC

(916) 721-8217
FAX: (916) 726-1905

Parallax, Inc.
6200 Desimone Lane, #69A
Citrus Heights, CA 95621

VISA MasterCard AMERICAN EXPRESS

California residents add sales tax.
Shipping: No charge for UPS ground.
$9.00 for UPS 2nd day, $18.00 for UPS next day.

Reader Service #151

1) Address line A9 is high and A8 is low. This corresponds to a hexadecimal base address whose most-significant digit is 2.

2) The next-most-significant hex digit is determined by comparing A7, A6, AS, and A4 with etude's base address jumpers.

3) The least-significant hex digit is always 0.

4) A3 is not decoded, and therefore the base address + 8 is redundant.

When A2 is low, the 8253 is selected. Conversely, when A2 is high the 8255 is selected. Specific registers within the 8253 and 8255 are accessed by Al and AO.

_InitializeSmartChips does just that. When the PC is booted or reset, all 8255 inputs are in the high-impedance state. This is a safe condition for étude until _InitializeSmartChips configures the programmable hardware.

Port A of the 8255 is programmed as an input for data from the cache or ADC to the PC bus via either DMA or I/O port reads.

Figure 2 shows the 8255's ports B and C, which are programmed as outputs. When reading/writing a 16-bit word in two 8-bit transactions, the order must be correct. The programmability of the 8253 allows the MSB to be read/written first, last, or not at all. We **have chosen** LSB/MSB to remain consistent with IBM's programming of other chips in the PC. This is critical when programming the PC's DMA controller and counter/timer.

ZeroAddressGenerator resets the address generator to zero. The cache is a sequential access device and may be considered a linear array of 8-bit data elements. A pointer into the cache selects a given element which then becomes available on the cache bus. Two operations may be performed on the pointer:

1) Increment the pointer via any of the mux input sources.

(See _MUXChannelSelect for more information.)

2) Reset the pointer to the "head" via this procedure.

When 8255 port C2 momentarily goes low, the address generator is

```
  _ADC_OutputBuffer (Enable);
  _AddressGenerator (Disable);
  _TTL_Buffer (Enable);
  _LoadSampleCounter (NumberOfSamples);     { SynchSampleCounter }
  _ZeroAddressGenerator;{ Must be done AFTER _LoadSampleCounter }
  _SetSampleRate (Frequency);                              { ditto }
  _AddressGenerator (Enable);                 { Software Trigger }
en&

procedure _GetAddress (var Data: var Page : byte;
                       var Offset : word);
var
  PhysicalAddress : longint;
begin
  PhysicalAddress := seg(Data);
  PhysicalAddress := (PhysicalAddress shl 4) + ofs(Data);
  Offset := PhysicalAddress and $0000FFFF;                 { 0..64K }
  Page := PhysicalAddress shr 16;                          { 0..15 }
end;

procedure _DoDMA (Page : byte; Offset, NumberOfSamples : word:
                  ResetPointer: boolean);
const
  StartingAddress = $06;
  ByteCount     = $07;
  Command       = $08; { write }
  Status        = $08;  { read }
  MaskRegister  = $0A;
  Mode          = SOB;
  FlipFlop      = $0C;
  CH3PageRegister = $82; {4 high-order bits of 20-bit address }
var
  QuickShutOff : byte:
begin
  NumberOfSamples := NumberOfSamples - 3;
  QuickShutOff := BitsC and $EF;              { Prepare to disable }
  port[ Mode ]:= $87;   { Block, Inc, No AutoInit, Write, CH3}
  port[ FlipFlop ] := $01;        { Any write will reset the FF }
  port[ CH3PageRegister]:=  Page;
  port[ StartingAddress ] := lo(Offset);
  port[ StartingAddress ]:= hi(Offset);
  port[ ByteCount ] := lo(NumberOfSamples);
  port [ ByteCount ]  := hi(NumberOfSamples);
  _MUX_ChannelSelect (IO-Read);
  if ResetPointer
    then _ZeroAddressGenerator;
  _DirectMemoryAccess (Enable);     { Assert PC bus DMA Request }
  _AddressGenerator (Enable);
  port[ MaskRegister ] := $03;        { Start the DMA transfer }
  repeat                                 { SPEED CRITICAL CODE }
  until (port[ Command ] and 8) = 8;
  port[ LatchC ]  := QuickShutOff;
  BitsC := QuickShutOff;
  _DirectMemoryAccess (Disable)    { release PC bus DMA request }
end;

procedure _CacheToMainViaDMA (var Data; NumberOfPoints : word;
                              ResetPointer : boolean);
var
  Page  : byte;
  Offset : word;
begin
  _GetAddress (Data, Page, Offset);
  _CacheReadWrite (Read);
  _ADC_OutputBuffer (Disable);
  _TTL_Buffer (Disable);   { Don't let Inhibit prevent read }
  _DoDMA (Page, Offset, NumberOfPoints, ResetPointer)
end;

function _ReadADC: byte;
var
  Dummy : byte;
begin
  _CacheReadWrite (Write);     { Don't let outputs conflict }
  _ADC_OutputBuffer (Enable);
  _MUX_ChannelSelect (IO_Read);
  _AddressGenerator (Disable);  _TTL_Buffer (Disable);
  Dummy := port [ CacheBus ];
  Dummy := port[ Cache&S ];{ First two conversions stale }
  _ReadADC := port [ CacheBus ];
end;                                                  (continued)
```

.isting 2-continued

```
procedure _ADCtoMainViaDMA (var Data; NumberOfSamples: word);
var
  Page : byte;
  Offset : word;
begin
  _GetAddress (Data, Page, Offset);
  _CacheReadWrite (Write); { Don't let outputs conflict }
  _ADC_OutputBuffer (Enable);
  _MUX_ChannelSelect (IO Read);
  _AddressGenerator (Disable);
  _TTL_Buffer (Enable);
  _DoDMA (Page, Offset, NumberOfSamples, true);
end;
```

listing 2-continued

cleared (all bits set to 0). This allows access to the "zeroth" element of the cache.

Note: Once the pointer reaches the "tail," a Terminal Count signal is generated and the mux is disabled. _ZeroAddressGenerator must be called to again enable the mux.

SETTING THE SAMPLE

_LoadSampleCounter sets the number of samples to be acquired. The sample counter receives pulses divided by 16 for two reasons:

1) It can only count at a maximum rate of 2.5 MHz.

2) The 20-bit address generator has a 1M-byte range, however the 16-bit sample counter has only a 64K-byte range. The sample counter is an Intel 8253 operating as a software-triggered strobe. In this mode, the strobe signal is not generated until N+1 clock pulses after the initial count is loaded. It also takes one clock pulse to load the current count into the 8253. Remember that one clock pulse to the sample counter is equivalent to 16 clock pulses to the ADC and address generator.

Sixteen-bit words in the 8253 are accessed by two 8-bit operations. The order (LSB/MSB or MSB/LSB) is set at initialization time. The minimum number of samples is 32, with increments of 16 thereafter.

SetSampleRate calls the procedure_MUX ChannelSelect for the appropriate input source and sets the 8253's variable divisor if necessary.

_SamplesRemaining polls the sample counter in the 8253 and returns the actual number of samples remaining divided by 16. The sample counter is a down counter operating in mode 0. It is a 16-bit register and must be read low-order byte first, followed by high-order byte.

When the sample counter reaches Terminal Count (zero), its output pin changes state, but the counter continues to decrement. The address generator's Cache Full signal will eventually halt data acquisition, however Acquisition Complete is signaled in software by counter "wrap around" (i.e., a count of 64K–1). You may wish to consider using an 8254 instead. This part is upwardly compatible with the 8253, and also provides the programmer with a way to directly test the state of the output pin.

_SetOutputCoding facilitates setting one of four possible representations for coding the analog input as a binary output: true binary, inverted binary, true two's complement, and inverted two's complement. The choice of output coding is based upon the data representation used (byte or shortint) and the adjustment of R3 (unipolar plus, unipolar minus, or bipolar). Most compilers use two's complement to store signed data, and true binary for unsigned data. You may avoid a conversion **by** choosing a data representation other than that normally used by the run-time system. For example, etude's demonstration software uses inverted binary rather thantruebinarybecauseof how the screen coordinates are mapped.

Software port PB3 (NLINV) and PB4 (NMINV) control output coding according to Figure 3.

PROGRAMS > 64K?

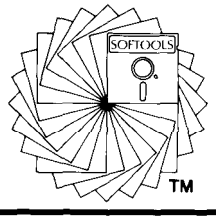
Auto-MMU Support Is The Answer.

SASM-Advanced Macro Cross Assembler
SLINK-Advanced Linker

Softools, **Inc. introduces a relocating macro assembler and linker package** that offers **many** features for the **embedded programmer at an** affordable price. It supports the **64180, Z80, 8085,** and 2280 processors.

SASM also **supports the 84180** MMU for automatic control of programs larger than 64K by making "long" calls into segments not mapped into the address space. It also includes many pseudo-opcodes for close compatibility with other assemblers. **SASM** accepts expressions that use operators common with other assemblers as well as C operator equvalents. SLINK is able to resolve any expression if SASM is unable to obtain a result. SASM includes a built-in MAKE facility which supports dependency file checks. It allows you to use one source file to generate a multi-module **library file.** In addition, SASM generates **full** source-level debugging information for each source file including the source name, include files, line numbers, public symbols, and local symbols.

SLINK output is compatible with In-Circuit Emulator (ICE) source-level debugging, and also generates binary or Intel HEX files and has the ability to divide output into multiple ROM image files. It supports named segments which may be up to 64K in length each, and may be linked to reside at one physical address and executed at another. Any banked or MMU controlled program requires this feature to locate code effectively. SLINK also allows the exclusion of physical address ranges in order to leave holes in the output file.



SOFTOOLS INC.
8770 Manahan Drive
Ellicott City, MD 21043
301-750-3733

ONLY $249

Reader Service #161

HIGHER-LEVEL FUNCTIONS

_FillCache-ViaADC sets the desired acquisition frequency and synchronizes the sample counter with the start of the data acquisition. The DB15 input Inhibit-Release\ (pin 8) must be released (not pulled low) for the acquisition to commence and progress. Notice that acquisition may also be inhibited via the _AddressGenerator procedure.

_GetAddress determines the starting address of a statically allocated arbitrarily sized array without knowledge of the array's extent. It normalizes a 20-bit address from segment and offset parts, and then separates thataddressinto theformexpected by the IBM PC's DMA hardware.

_DoDMA programs the PC's DMA hardware. Due to the segmented architecture of 80x86 processors, the 8237 DMA controller only recognizes 64K byte pages. A 1M-byte address space, therefore, has 16 (2^4) such pages. Each DMA channel has a corresponding 4-bit page register. (In the PC, DMA channels 0 and 1 share a common page register, but this is not a handicap since channel 0 can only be used for refresh.)

The following example shows how the value of the DMA page register is computed, and illustrates a limitation inherent in the architecture of the 80x86. Let's say the first element of the storage array is located at segment 1F00H, offset 2000H. Both segment and offset values are 16-bit unsigned integers. In order to make a 20-bit physical address, the segment is left-shifted by four and added to the offset, resulting in the absolute address 21000H. In this example, the page register would get the value 2, and the starting address for the DMA transfer is 1000H. Now suppose a transfer of FEOOH data points is requested. This causes a problem because the starting address plus the number of bytes to transfer can't result in a carry to the page register since there is no provision for updating the page register in the middle of a DMA transfer.

In general such a limitation is not a drawback; we just request two separate DMA transfers and update the page register **prior** to the second transfer. However, when using the non-cached mode, the number of points we can acquire is limited not by the 64K-byte transfer size of the 8237 itself, but by where the compiler puts the data array, Therefore, we need to dynamically ascertain the location of a free page in memory or explicitly force the compiler to do so. One problem with the static allocation scheme is that it doesn't take into account any TSR (Terminate and Stay Resident) programs that may have been previously loaded by DOS.

Finally, DMA transfers are one byte more than requested (this is so a full 64K bytes can be transferred with a 16-bit request) and the constants used are register locations for DMA channel 3.

Base Address (Hexadecimal)	Jumper Settings A7	A6	A5	A4	Possible Conflict
200	0	0	0	0	Game Port
210	0	0	0	1	Expansion Unit
220	0	0	1	0	Reserved by IBM
230	0	0	1	1	Reserved by IBM
240	0	1	0	0	Reserved by IBM
250	0	1	0	1	Suggested for étude
260	0	1	1	0	
270	0	1	1	1	LPT2
280	1	0	0	0	
290	1	0	0	1	
2A0	1	0	1	0	
2B0	1	0	1	1	
2co	1	1	0	0	
2D0	1	1	0	1	
2E0	1	1	1	0	
2F0	1	1	1	1	COM2

Figure 1—*étude* may occupy any of the pot-f locations shown here. but a number are *already occupied.*

_CacheTo-MainViaDMA sets up étude for a DMA transfer and calls GetAddress and DODMA. Since the DMA controller only recognizes 64K-byte pages, the cache address pointer must remain where it was last, and another DMA transfer must commence in order to retrieve more than 64K bytes from etude (this is not a consideration with the 4K-byte version of etude).

_ReadADC demonstrates operation in one of the noncached modes. It is a slow and easy method of obtaining data in "real-time." The ADC itself has a two-conversion pipeline latency before spitting out "fresh" data. This procedure first disables the address generator so the Cache Full signal can't halt the process. The Inhibit-Release\ line (DB15 pin 8) has no effect in this mode.

ADCtoMainViaDMA demonstrates operation in the other non-cached mode. Data is converted as fast as the DMA controller can write it to main memory, which allows up to 65,536 samples before the DMA controller must be reprogrammed. It may be useful for applications which require more than 4K bytes of data, albeit at slower acquisition rates.

The sample frequency is directly related to the DMA performance of your computer. It may be measured on pin 5 (Timebase Output) of the DB15, however this signal is only available while a DMA transfer is in progress (typically only milliseconds for a 64K-byte transfer). Fortunately, it is easy to estimate the sample frequency for DMA acquisitions. Simply connect a variable-frequency waveform generator to etude's analog input and adjust it so that one period of the waveform exactly fills the display

screen (512 points). The DMA sample frequency then equals 512 times the waveform generator frequency (for our AT clone this worked out to about 1.8 MHz). The exact DMA frequency is related to the 8237's clock period, programmed mode, and the number of wait states inserted for 8-bit device I/O. The 8237 usually transfers one byte per three clock periods; however, when it is programmed for compressed timing, the transfer rate increases to almost one byte per two clock periods since address bits A15 thru A8 need be updated only every 256th byte. If the time between samples must be constant (e.g., FFT post processing), the compressed mode of operation is not useful. You may ask, "Doesn't refresh cause the same sort of problem?" The answer is no. After recognition of DMA service by any channel, the other channels are prevented from interfering until that service is completed. Since refresh uses DMA channel 0, there is no interference.

Port	Mnemonic	Function
PB7	S0	MUX Input Select
PB6	S1	MUX Input Select
PB5	s2	MUX Input Select
PB4	NMINV\	Output Coding Format
PB3	NLINV\	Output Coding Format
PB2	DMA\	DMA Enable
PB1	RESERVED	
PB0	RESERVED	
PC7	OE\	ADC Output Enable
PC6	RD\	Cache Read
PC5	WR\	Cache Write
PC4	INHIBIT\	Stop Addrs Gen From Incr
PC3	RESERVED	
PC2	CLR\	Zero Address Generator
PC1	RESERVED	
PC0	ENBUF\	Enable Ext TTL I/O Buffer

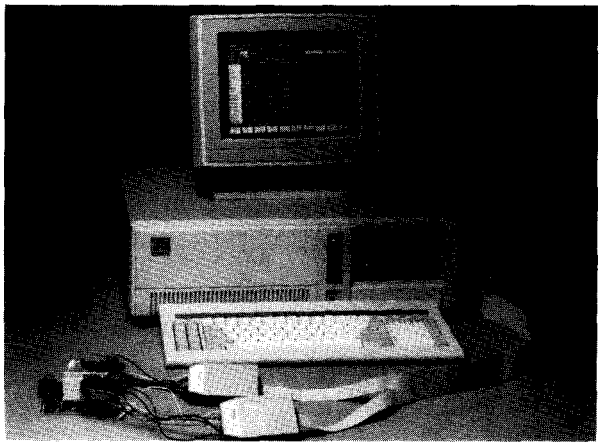
Figure *P-Ports B and C of étude's 8255 are used to control various aspects of the board.*

A FEW TIPS

That pretty much sums up the low-level software. When you've built the hardware, this software provides a foundation for any number of special analog-to-digital applications. We have developed several applications, and the editors of CIRCUIT CELLAR INK would like to hear about any applications that the readers develop. The door is wide open for application articles in future issues.

When you get ready to build etude, here are a few tips which will help with the construction:

PC-Based Logic Analyzers



Sophisticated Logic Analysis at Unsophisticated Prices

*ID160 (50 MHz) for $695

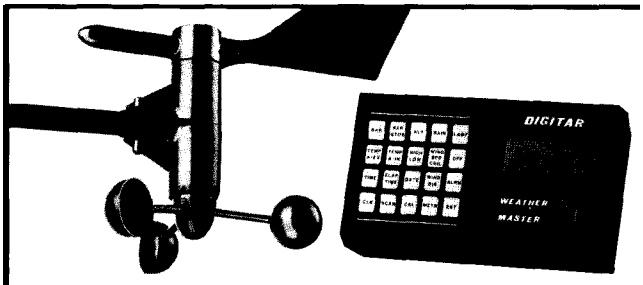
*ID161 (100 MHz) for $895

•50 MHz or 100 MHz Sampling • 8K Trace Buffer • 32-channel Operation *Multi-Level Triggering *State Pass Counting *Event Timer/Counter *Performance Histograms *Hardcopy Output *Disassembles popular 8-bit micros *and much more !

*30 Day Money Back Guarantee

INNOTEC DESIGN, INC.
6910 Oslo Circle, Suite 207
Buena Park, CA 90621
Tel: 714-522-1469 FAX:714-527-1812

Reader Service #133



PROFESSIONAL-QUALITY WEATHER STATION USED BY RADIO AND TV METEOROLOGISTS. NOW ONLY $285

We've sold Digitar Weather Master weather stations to radio and TV stations from Bangor, Maine to central Borneo because we offer sophisticated, professional-quality weather monitoring at an incredibly low price. Our state-of-the-art microcomputer technology gives you unprecedented ability to monitor, record and compute weather data. The Weather Master includes a computer, electronic barometer, remote precision wind direction vane, wind speed sensor, external temperature sensor, programmable alarms, mounting hardware and 40' of cable – all for only $2851 Features include:

- Barometer (w/memory)
- Altimeter (w/alarm)
- Wind Speed (w/alarm)
- Wind Direction
- Windchill Factor
- Inside Temp. (w/alarm)
- Outside Temp. (w/alarm)
- Min/Max Temp. Record
- Wind/Gust Record
- Daily/Yearly Rain
- Time of Day (w/alarm)
- Elapsed Time
- Four-Year Calendar
- Metric and Standard
- Programmable Scan
- Portable • Backlit LCD

DIGITAR WEATHER MASTER WEATHER STATION ONLY 5285.001
ORDER TODAY: 1-800-678-3669
14-DAY MONEY BACK GUARANTEE 1-YEAR WARRANTY
Self-emptying rain guage -$49.95(optl). Add $5.00 for shipping, CA residents add sales tax
FAX 1-415-732-9188 • M/C and VISA
DIGITAR . 3465 DIABLO AVE. . HAYWARD, CA 94545

Reader Service #119

PB3 NLINV\	PB4 NMINV\	output Coding
0	0	Inverted Binary
0	1	Inverted 2's Complement
1	0	True 2's Complement
1	1	True Binary

NLINV\ stands for: invert all but the most significant bit.
NMINV\ stands for: invert most significant bit.

Figure 3—*The form of output coding used by étude is determined by NLINV\ and NMINV\.*

*TRW has dropped the -1 designation; all THC1068s now run at 25 MHz.

•Yes, the converter gets very hot. It dissipates 1.6 watts!

•The converter is available from Hall-Mark, Hamilton Avnet, and Silicon Alley.

•If you build étude from the kit, please note that the orientation of the converter is opposite that of the other chips. Follow the markings on the circuit board silkscreen.

*The DB15 is available cheaply from JDR Microdevices. The shell grounding tabs which run under the mounting holes of this part should be broken off prior to installation. ✣

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

Dick Hubert is one of the founders of A.P.P.L.E. Coop, is a partner in Silicon Alley, and has been involved with microcomputers for ten-to-the-sixth years.

étude is available from:

Silicon Alley, Inc.
P.O. Box 59593
Renton, WA 98058
(206) 255-7410

both in kit and finished forms. The kit contains: PC board, PAL, 4 KB of SRAM, manual, and software for $99.00 (introductory price). An assembled and tested version sells for $495.00. PALs are available for $5.00

Extended cache versions (128,256, 512 KB, and 1 MB) are available from:

Rapid Systems, Inc.
433 North 34th
Seattle, WA 98103
(206) 547-8311

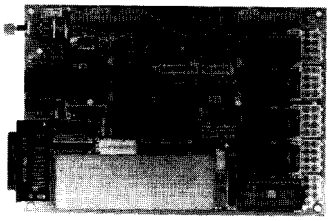
IRS

213 Very Useful
214 Moderately Useful
215 Not Useful

New, Improved!

SIBEC-II

The ideal solution for embedded control applications and stand-alone development.



- *Intel 8052AH BASIC CPU*
- *Serial printer output and* **5, 8** *bit I/O ports*
- *5 in.² prototyping area*
- *Memory: 8K RAM, expandable to 128K*
- *Power requirements: 5V.DC @ 300 ma. only*
- *PROM progammer; ZIF socket for 2764 or 27128 EPROM*
- *Interrupt handling capability*
- *Built to exacting standards and warranteed*
- *Still only $228.00 including documentation (quantity 1)*

Inquire about our PDK51 8051-8052 product development kit for the IBM-PC/XT/AT: $595.
Our BXC51 8051/8052 BASIC compiler: $295.

Call now! ***603-469-3232***

Binary Technology, Inc.
Main Street • PO. Box 67 • Meriden, NH 03770

Reader Service #106

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

Twenty new *easy-to-build, cost effective*, fun projects including:
a touch-tone interactive message system...the world's smallest 1200-BPS modem...a turnkey bulletin board system...an audio and video multiplexer. . . an analog to digital converter...a serial Eprom programmer...and much more.
256 pages, 100 illustrations.

CIARCIA'S CIRCUIT CELLAR

Volumes I-VII $21.95 each

Call
(203) 875-2751 to order

Reader Service #111

The Furnace Firmware Project

Process Control on the Home Front

FIRMWARE FURNACE

Ed Nisley

So far, the Firmware Furnace has explored interesting firmware snippets, presented some small projects, and expounded the odd tutorial topic. Now it's time for a Big Project, because I need some Furnace Firmware.

The task is to monitor the fuel oil used to heat my office at home. The standard method uses the ratio of the office area to the total house floor space, but that isn't accurate because we heat the rest of the house only during the evening. I think I can do a much better job with a small computer, a few sensors, and, ah yes, a little firmware...

The computer will need the usual display, keypad, serial I/O, and sensor programming, so, starting with this column, I will explore each area in turn and present some fairly general-purpose firmware to control each device. You get **some** useful new tools, I get a useful new gadget, and we both learn new things along the way.

But first I must explain how an oil heating system **operates;** while it's not hard to understand, there is a lot of mystery surrounding the hardware, particularly among folks in Paradise where homes are always warm. You may notice similarities between my oil furnace and industrial process control: after all, we're dealing with pumps, motors, relays, switches, and (worst of all!) fluids

IN HOT WATER

The whole point of a home oil furnace is to transfer heat energy from a fire in the basement to the rest of the house, using hot water pumped through baseboard radiators. Most houses have several zones, each with a thermostatic switch that closes when the room air chills below the zone's temperature setpoint.

Figure 1 sketches the essential heating control circuitry. When the thermostat closes, the zone valve's motor opens a valve so hot water can flow into the radiator. A limit switch

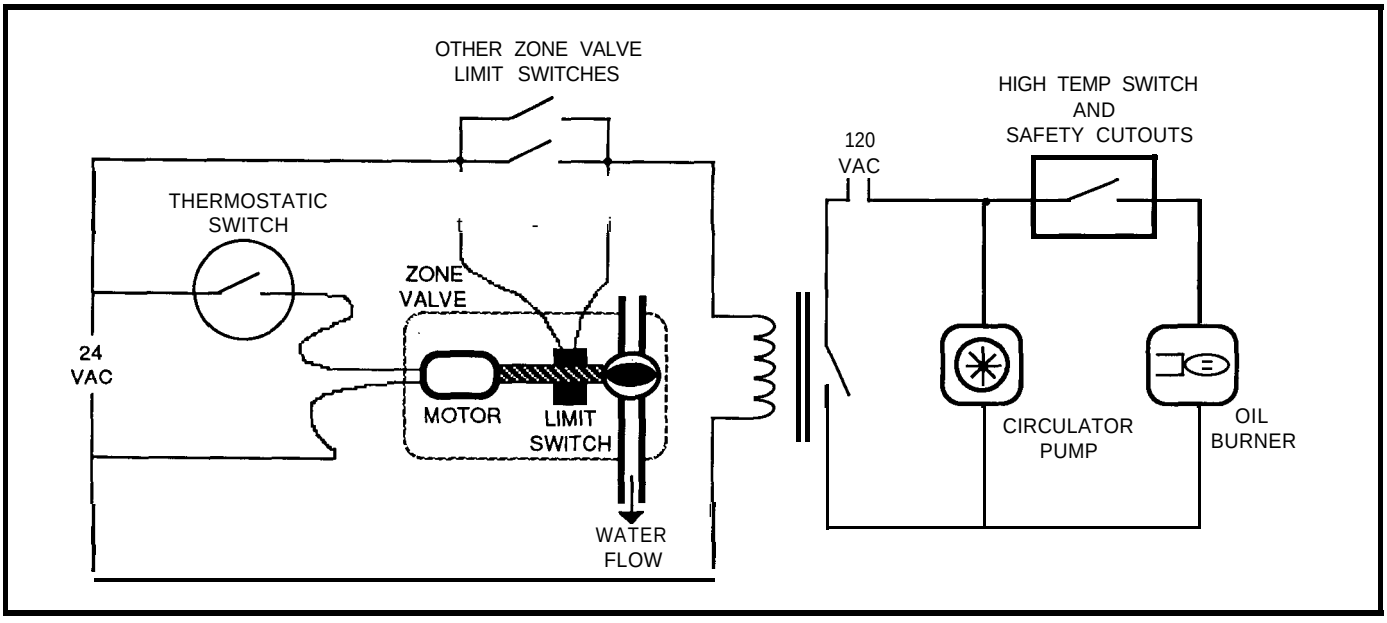


Figure 1—*Thermostats, valves, limit switches,* **controllers,** *and pumps make up the bulk of the furnace circuit. The entire setup is basically a scaled-down industrial control process.*

8031 µController Modules

NEW!!!

Control-R II

√ Industry Standard 8-bit 803 1 CPU
√ 128 bytes RAM / 8 K of EPROM
√ Socket for 8 Kbytes of Static RAN
√ 11.0592 MHz Operation
√ 14/16 bits of parallel I/O plus access to address, data and control signals on standard headers.
√ MAX232 Serial I/O (optional)
√ +5 volt single supply operation
√ Compact 3.50" x 4.5" size
√ Assembled & Tested, not a kit

$64.95 each

Control-R I

√ Industry Standard 8-bit 803 1 CPU
√ 128 bytes RAM / 8K EPROM
√ 11.0592 MHz Operation
√ 14/16 bits of parallel I/O
√ MAX232 Serial I/O (optional)
√ +5 volt single supply operation
√ Compact 2.75" x 4.00" size
√ Assembled & Tested, not a kit

$39.95 each

Options:

MAX232 I.C.	$6.95 each
6264 8K SRAM	$10.00 each
8052BAS IC CPU	$25.95each

Development Software:

PseudoSam 5 1 Software ($50.00)
Level II MSDOS cross-assembler.
Assemble 803 1 code with a PC.

PseudoMax 51 Software ($100.00)
MSDOS cross-simulator. Test and debug 803 1 code on your PC!

Ordering Information:

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

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

Reader Service #116

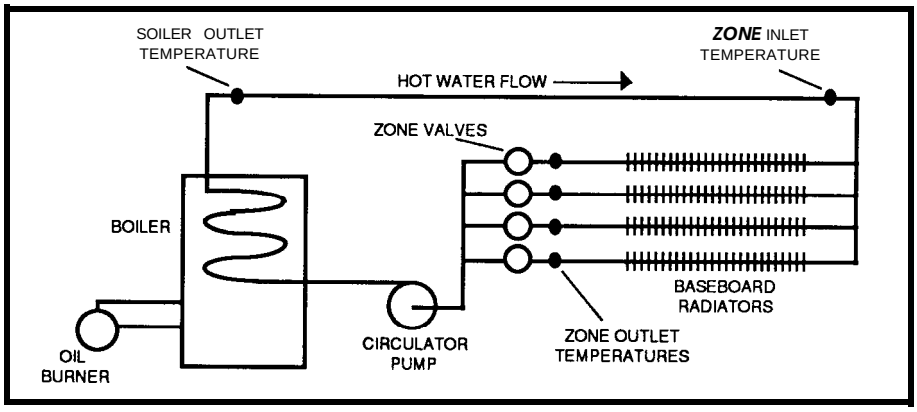


Figure 2—*My system* **burns** *fuel-oil* ***and*** *delivers heat to four thermostatically controlled zones. While there are subtle differences in individual installations, all fossil-fuel heating systems will be somewhat similar to this.*

closes when the valve is fully open, which tells the furnace controller to begin providing heat.

A controller relay activates the circulator pump and fires the oil burner. The circulator runs whenever any zone is active, but the burner can be turned off by the high-temperature limit switch that monitors the boiler hot water temperature. Several safety interlocks can also shut the burner off if they detect abnormal conditions like no flame or high air temperature.

Notice that the thermostats and zone valves run on 24 VAC, while the circulator and burner use straight 120 VAC. Home heating control has barely moved into the solid-state world, let alone begun to use digital logic. Gettinginformationout of this system will be more complex than just sticking a wire on a screw terminal!

Our house hasfourheatingzones: upstairs, downstairs, my office, and the hot water heater. Plumbing a hot water heater as a heating zone is unusual, but the previous owners did this when the domestic hot water coil in theboiler developed hard-water arteriosclerosis. For our purposes, it's just another zone with a thermostat and zone valve. Figure 2 shows the plumbing layout and the location of the temperature sensors that measure the heat delivered to each zone.

There are, of course, variations on the theme of heating. Your system may use hot air instead of hot water, burn natural gas instead of fuel oil, or have one zone or five. The furnace controller, in particular, may have different algorithms to control the burner. All fossil-fuel home heating systems are basically similar, so I'll leave it to your ingenuity to figure out what goes on in your basement.

IN HOT WATER

The baseboard radiators **along** the outside walls of each zone transfer energy from circulating hot water to room air. Natural convection moves warm air from the radiators into the room, replacing it with cold air drawn from near the floor. The hot water cools as is proceeds along the radiators and returns to the furnace boiler for another dose of heat.

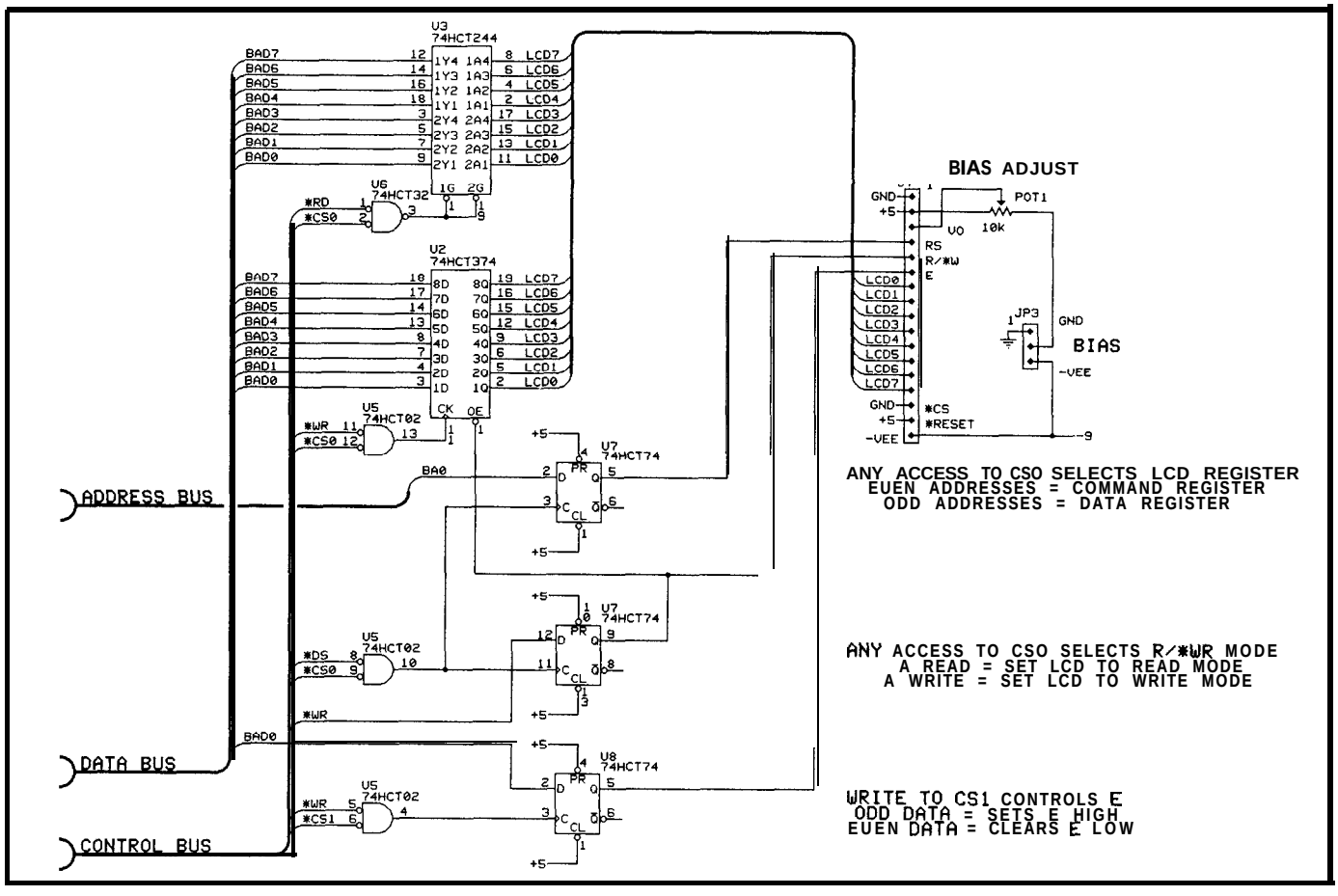
Although calculating energy transfer in heat exchangers from first principles is a difficult problem (well suited for Professional Engineering qualification exams!), measuring the

Figure 3—*The* **energy** *delivered to a heating zone is a function of the temperature differential and the flow rate through the baseboard.*

$$\frac{BTU}{sec} = \left(\frac{ft^3}{sec}\right)\left(\frac{lbm}{ft^3}\right)\left(\frac{BTU}{lbm°F}\right)\left(T_{in} - T_{out}\right)$$

FLOW RATE — DENSITY — SPECIFIC HEAT — TEMPERATURE DIFFERENCE

FOR WATER:
DENSITY = 62.4 lbm/ft³
SPECIFIC HEAT = 1 .O BTU/lbm°F



Schematic 1— Jhe *portion of the RTC-LCD board containing the* LCD *interface uses* individual latches to *control* the LCD's *register select, read/write, and E inputs.*

transfer in an existing exchanger is straightforward. For a given fluid, the heat transferred per unit time is directly proportional to the fluid mass flow rate and the inlet-to-outlet temperature differential. Figure 3 presents the equation and constants applicable to ordinary water.

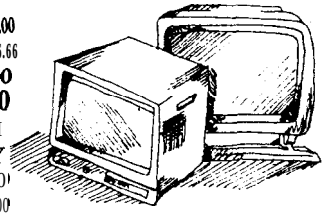
Finding the temperature differential is easy enough: two sensors, a calibration curve, and digital subtraction will suffice. But measuring water flow is another matter entirely, as water flow meters rated for 200-degree service are not cheap. Because each zone's flow depends on which other zones are active, my simple system needs four flowmeters to calculate the heat delivered to each zone.

Fortunately, thereisaneasier way. Theflowrateisinverselyproportional to the flow time through the zone; the pipe volume is fixed, so measuring the elapsed time gives the rate. Even better, the outlet temperature sensor

FOW COMMUNICATIONS
CAD MONITORS SPECIALS!
508-485-1144
FAX 508-481-7222
BBS 508-460-9203

LOWBUDGETSPECIALS

19" COLOR SUPER VGA MONITOR
LIKE NEW 6 MONTH WARRANTY **$800.00**
USED 3 MONTH WARRANTY $666.66
USED WITHOUT CASE ..... $300 **TO $500.00**
19" COLOR SONY 1280 X 1024 ....$1600.00
THESE SONY TRINITRONS HAVE A FULL 6 MONTH WARRANTY PARTS AND LABOR.LIMITEDQUANITY
16" IKEGAMI 64KHZ 1280X1024 NEW $899.00
16" PANISONIC 64KHZ USED $599.00
14" IKEGAMI TTL VGA CHASIS $249.00
NEW WITH 1 YEAR WARRANTY



19" PHILLIPS 1024X800 48 KHZ GRAY SCALE **$350.00**
NEW **1 YEAR** WARRANTY - MAY BE ORDERED FOR VGA AT NO EXTRA CHARGE. WHEN USED IN VGA MODE THE MONITOR WILL RUN **800 X** 600 **X 256** GRAY **SCALE** OR **1024X768X16 GRAY SCALE ONLY**
CALL US ABOUT OUR **LARGE** VARIETY OF GRAPHIC CARDS !
194 Main ST. Marlboro,MA 0 1752

Reader Service # 126

8031/51 Tools

8031 In-Circuit Emulation

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

8051 Simulation

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

8031/51 Single Board Computer

A fast and inexpensive way to implement an embedded controller. 8031/32 processor, 8+ parallel I/O, up to 2 FiS232 serial ports, +5 volt operation. The development board option allows simple debugging of 8031/51 family programs. **$99**ea

Prototyping System

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

Call us for your custom product needs.

Other products available:

MyGAL - GAL Programmer $199

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

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

(619) 566-l 892

Reader Service x131

indicates when the first hot water reaches the end of the zone, so we don't need any additional hardware.

The program can build a table of zone flow rates "on the fly" as various zones are activated if it knows the pipe length (and thus the pipe volume) of each zone. The code can measure a flow rate only when the zone is first turned on, because only then will there be a sharp temperature difference between the cool water already in the radiator and the hot water just starting to circulate.

```
WRITE CHARACTER DATA TO LCD CONTROLLER:
        XBY (0E081h) = DATA   : REM DATA TO U2, CLR RD/WR, SET RS
        XBY (0E090h) = 1      : REM TURN EON
        XBY (0E090h) = 0      : REM TURN E OFF

WRITE INSTRUCTION TO LCD CONTROLLER:
        XBY (0E080h) = DATA   : REM DATA TO U2, CLR RD/WR, CLR RS
        XBY (0E090h) = 1      : REM TURN EON
        XBY (0E090h) = 0      : REM TURN E OFF

READ CG OR DD DATA FROM LCD CONTROLLER:
        dummy = XBY (0E081h) : REM SET RD/WR, SET RS
        XBY (0E090h) = 1     : REM TURN EON
        data = XBY (0E081 h) : REM READ DATA
        XBY (0E090h) = 0     : REM TURN E OFF

READ BUSY BIT FROM LCD CONTROLLER:
        dummy = XBY (0E080h) : REM SET RD/WR, CLR RS
        XBY (0E090h) = 1     : REM TURN E ON
        data = XBY (0E080h)  : REM READ BUSY BIT (BIT 7)
        XBY (0E090h) = 0     : REM TURN E OFF
```

Figure 4—*The* RTC-LCD uses *latches to control the HD44780 Register* **Select,** *Read/Write, and Enable inputs. These BASIC examples assume a base address of E080h for the RTC-LCD.*

I'll go into more detail on the data measurement and recording hardware and program in subsequent columns. First 1 must get some computer hardware on the air.

HARDWARE HOOKUP

One's first impulse on starting a project is to whip out the soldering iron and build some hardware. But the fact of the matter is that you generally don't need a full-custom system, particularly when you're trying to do something as fundamental as measuring a few temperatures.

Jeff has already mashed nearly everything I need for this project onto RTC52-sized boards: the display and keypad interfaces, analog-to-digital converters, a whole computer (an8052 is a computer, honest), AC power

```
; Send command or data to LCD controller
;F0 set -> d a t a   b y t e
; F0 clear -> command byte
; Crunches DPTR
; "Force" entry bypasses the busy test

            SEG         UtilCode
LCDSendByte PROC
            CALL        LCDWaitBusy     ; make sure it's ready

LCDForceByte EQU        $               ; bypass busy test

            PUSH        ACC             ; save data
            GetCWord    DataAddr        ; point to data port
            JB          F0,L?latch      ;  is it a command?
            GetCWord    CmdAddr         ;  yes,  set RS flag
L?latch
            POP         ACC             ; recover data
            NOVX        @DPTR,A         ; latch the data byte

            GetCWord    StrobeAddr      ; point to strobe port
            MOV         A,#1            ; pulse the strobe
            MOVX        @DPTR,A
            CLR         A
            MOVX        @DPTR,A
L?done
            RET
LCDSendByte ENDPROC
```

listing 1 -This *routine sends a byte to the HD44780 LCD controller. If the F0 CPU flag is set, the byte goes into Display RAM as a character; otherwise, it is treated as a controller instruction. A separate entry point bypasses the normal BUSY wait for bytes sent during the reset sequence when the controller is in an unknown state.*

I/O, and so forth. Rather than design new hardware, I'll stack some RTC boards and concentrate on the code.

The hardware will include an LCD panel, a membrane keypad, analog voltage inputs, and a few AC power inputs. I'll use the venerable 8052, but much of the code we'll be exploring here will be written in assembler.

An RS-232 link will provide remote control and data reporting capabilities so I don't have to write things down on paper. I plan to make the keypad and display work in parallel with the serial link, so a minimal system could omit the LCD and keypad.

If you want to build a similar system from scratch, you can probably fit everything onto a single board. I will provide schematics of the interestingparts of the hardwareand some test programs to check out the system. However, you must put your hands on the keyboard and soldering iron, because this isn't a "finished kit" project by any means.

SOFTWARE STRUCTURE

The Furnace Firmware code will appear piecemeal during the next several columns, so I should describe how all the parts fit together. Thecode structure allows you to extract useful routines for your own projects without dragging along a lot of excess baggage.

I plan to write the overall monitoring program in C to find out how well a high-level language fits in a microcontroller. As the Circuit Cellar BBS regulars among you already know, I'm not convinced this is Entirely A Good Thing, but I'll give it a fair shake. If all else fails, there's always BASIC-52 in ROM!

However, most of the code you'll see here will be assembly language display, keypad, and other hardware drivers. While C can certainly handle much of these functions, that would mean you'd need a C compiler to get much benefit from this project; that's definitely a Bad Thing. The driver test programs will be free-standing assembler routines that you can burn into an EPROM, most of the hardware checkout code is written in BASIC-52, and we'll ease into C only near the end.

Because the actual hardware I/O addresses are jumper selectable, all address "constants" are grouped in a segment near the end of the EPROM. If you need to change an address to match your hardware, you can modify a byte or two in the EPROM image without having to reassemble the program.

```
; Verify that LCD is ready for new data
; Returns C set if busy, C clear if OK to send

            SEG       UtilCode
LCDTestBusy PROC
            PUBLIC    LCDTestBusy

<<< startup code omitted >>>

            GetCWord  CmdAddr
            MOVX      A,@DPTR          ; set R/-Wr latch high

            GetCWord  StrobeAddr       ; turn strobe on
            MOV       A,#1
            MOVX      @DPTR,A
            GetCWord  CmdAddr          ; fetch the busy bit
            MOVX      A,@DPTR
            MOV       C,ACC.7          ; set return flag
            JNB       SimMode,L?norm   ; simulating?
            CLR       C                ; yes, force OK
L?norm
            GetCWord  StrobeAddr       ; turn strobe off
            CLR       A
            MOVX      @DPTR,A

<<< takedown code omitted >>>

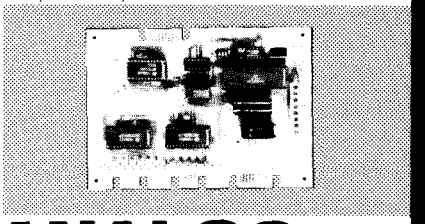
            RET
LCDTestBusy ENDPROC
```

Listing 2—*The hardware shown in Schematic 1 allows you to read data from the LCD controller. This routine reads the BUSY bit and clears the CPU's Carry flag when the HD44780 is ready for the next instruction or data byte.*

RELAY INTERFACE

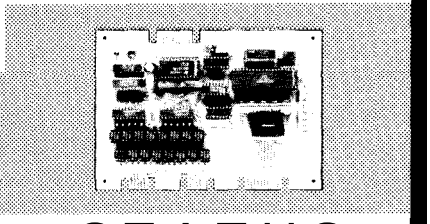
AR-16 Relay Interface ..... $ 89.95
Provides software control for 16 external relays, expandable to 128 relays with EX-16 expansion cards Plug-in relay cards, power relays and latched relays with status available May be connected directly to serial printer port or RS-232.

EX-16 Expansion Card $59.95
(16 channel)



ANALOG TO DIGITAL

ADC-8 (8 channel)' ..... $ 89.95
ADC-16 (16 channel)' ..... $ 99.95
Input temperature, joystick movement, voltage, pressure, energy usage. light levels etc. Voltage input level is adjustable (0 to 5 volts typical) Connects to W-232 or RS-422 ports.



STATUS INPUT

STA-8 (8 channel)' ..... $ 99.95
STA-16 (16 channel)'..... $119.95
STA-16-TT (touch tone input)' . . $159.95
Input on/off status of switches, relays, thermostats, security devices, smoke detectors, pressure switches and hundreds of other devices. Touch Tone input decodes all standard telephone touch tones and may be connected directly to a telephone line using an inexpensive coupler. Connects to RS-232 or RS-422 ports.

'Inputs are expandable to an additional 126 Status inputs or 16 analog Inputs. Add up to 112 relay outputs using EX-16 expansion cards.

. FULL TECHNICAL SUPPORT...provided over the telephone by our staff. A detailed technical reference manual is provided with each order including software examples in Basic and Assembly Language

• Engineered for continuous 24 hour industrial applications.

• Use with IBM and compatibles, Tandy, Apple and most other computers with RS-232 or W-422 ports. All standard baud rates and protocols may be used (50 to 19,200 baud) default is 9,600 baud 8 data bits, 2 stop bits, no parity. Use our 800 number to order free information packet.
Technical Information (614) 464-4470.

24 HOUR ORDER LINE (800)-842-7714
Visa Mastercard American Express COD

ELECTRONIC ENERGY CONTROL, INC.
380 South Fifth Street, Suite 604
Columbus, Ohm 43215

Reader Service #121

Similarly, all the program variables are grouped near the end of the External RAM. Although irrelevant in C, this will simplify using the routines with BASIC-52, as your startup code can set MTOP just below the variables. Grouping all the variables together makes dumping them to the console easier, whether you're using BASIC or an 8031 debugger.

The LCD driver, which I will discuss below, provides a simple TTY-style LCD interface. It handles the CR, LF, BS, and FF control characters, plus standard ASCII text. Your program can also set the cursor to any character cell in the display, which may be all you need for your application!

The keypad driver, coming in the next issue, will wrench a full alphanumeric keyboard with function keys and some punctuation from 16 membrane switches. All the keys feature the holdoff and repeat action usually found in fancier keyboards. Obviously, this is far more than I need for the Furnace Firmware project, but the extra functions aren't that hard and may come in handy on your project.

Next, the temperature sensorsand AC voltage conversion will get some attention. These may require some custom circuitry on an RTC-PROTO board, unless Jeff has something **clever** up his sleeve. Because the Furnace Firmware must measure time intervals and record events by wall-clock time, I'll look into the real-time clock option on the RTCIO board, and perhaps add some nonvolatile RAM to hold calibration constants.

Finally, I'll wrap everything up with a C program that will collect data and report results. One goodie will be an ANSI driver for the LCD panel, so the same control sequences will work on either the local display or an ANSI terminal (pronounced "PC") hung on the serial port. Despite my best efforts, the LCD won't handle the ANSI color change commands correctly, but the cursor positioning sequences will work!

DISPLAY INTERFACE

The Furnace Firmware display is one of those ubiquitous LCD panels driven by the Hitachi HD44780 controller. Last year, in CIRCUIT CELLAR INK #8, I presented some code to check out these panels using a PC, so you should be familiar with how they work. That C code was slow enough that timing parameters weren't an issue, but now that we're on the firmware level we must pay attention to details.

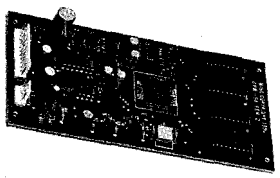
I plan to use a 4-line x 20-character LCD panel, but the code refers to EPROM constants that define the actual number of rows and columns. Another EPROM table stores the starting address of each line (in the HD44780's display RAM), which eliminatesa lot of tricky code required to do even a simple linefeed. The code handles 4x20, 2x20, 1x16, and even 1x8 displays.

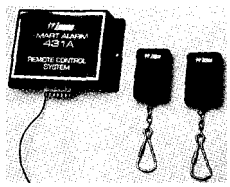
The LCDDEMO.BAS program will help you get your hardware working. It fills the display with characters, then rewritesthemdisplacedonecharacter to the left. This will exercise the outgoing data path and latches, but, since BASIC-52 will never see the HD44780 being busy, that logic isn't verified.

ELECTRONICS 1 2 3

A Division of MING E&P Inc.

1. EXCLUSIVE ITEMS, GOOD PRICE.
2. UNIQUE ITEMS, BETTER PRICE
3. POPULAR ITEMS, BEST PRICE,





DIGITAL VOICE RECORDING MODULE
- Long recording up to 2 mins
- 10 bit A/D, 32 Kbps
- Auto repeat feature
- 1 Mb DRAM (Expandable to 4Mb)
- Applications are unlimited

MING DVM-58 $49.99

RF REMOTE CONTROL SYSTEM
- 4096 Digital coded number
- 2 tiny transmitters
- Receiver has dry contact relay output
- Confirming signal output

ZEMCO SA432 $49.99

PASSIVE **INFRARED** DETECTOR
- Security industry quality
- Very compact & reliable
- SMT w/best RFI immunity
- Cover area 50' x 50

ROKONRT RK3000 $49.99

SYNTAX PROTOTYPING PCB
- All types of BUSs
- High quality FR-4 material
- Get your job done quickly

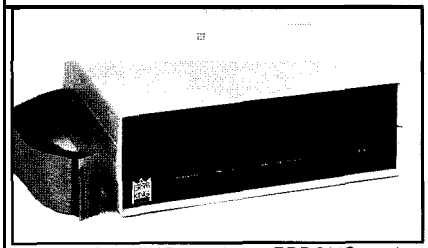
Please Call forFree Catalog

1-(800) 669-4406

977 S. Meridian Ave., Alhambra, CA 91803
Tel: (818) 281-4066 Fax: (818) 576-8748
VISA & MASTER CARD ACCEPTED

Reader Service #146

STOMP OUT EPROM MADNESS



The PROM KING emulates EPROMS saving both time and money during your development cycle. Programmable in seconds via your PC printer port or any computer RS232 port, it can emulate most 27xxx devices.

- 8K-8M bit devices
- High speed download:
  - -Universal RS232
  - -PC printer port
- Menu driven software
- Battery backup
- 8-256 bit downloads
- Easily expandable:
  - -4 EPROMS per unit
  - -Up to 8 units
- Also programs like a real EPROM

$599 for 150nS units with 256K bits
Ask for pricing of other options

Made in USA by:

TRAXEL LABS INC.

Box 239 •**RONKONKOMA**, NY -11779
516-737-5147

Reader Service #165

```
;-------
; Send character from ACC to LCD, handle control chars

LCDSendChar  PROC

             PUSH      ACC             ; save input char
                                       ;   around tests
;--- CR (carriage return)
             CJNE      A,#CR,L?notcr ; carriage return
             POP       ACC             ; discard char
             MOV       B, #0           ;   to first column
             GetXData  CurrentRow ;   of current row
             CALL      LCDSetCursor    ; force first column
             GetXData LCDControls    ; want LF, too?
             LJNB      ACC.LCD_CRLF,L?done ; nope
             MOV       A,#LF           ; yup, force LF 6 fall
                                       ;  through!
             PUSH      ACC             ; fake saved data
L?notcr

;--- LF (line feed)
             CJNE      A,#LF,L?notlf ; linefeed
             POP       ACC             ; discard char
             GetXData  CurrentCol      ; save current column
             PUSH      ACC
             GetCData  VisibleRows
             MOV       B,A
             GetXData  CurrentRow      ; step the current row
             INC       A
             CJNE      A,B,L?lf1       ;  C set if current
                                       ;    < visible
L?lf1        JC        L?lf2           ;  . ..thus no scrolling
             DEC       A               ; back up to last row
             PUSH      ACC             ; save around scroll
             GetXData  LCDControls     ; scrolling enabled?
             JB        ACC.LCD_VSCROLL,L?lf2a
             POP       ACC             ; no, discard saved row
             CLR       A               ;  ... return to top row
             SJMP      L?lf2           ;  and avoid scrolling
L?lf2a       CALL      LCDScrollUp
             POP       ACC             ; row back
L?lf2        POP       B               ; to new row & existing col
             SJMP       L?setcurs
L?notlf

;--- FF (form feed)
             CJNE      A,#FF,L?notff ;  form feed is easy!
             POP       ACC             ; discard char
             CALL      LCDClear
             SJMP      L?done
L?notff

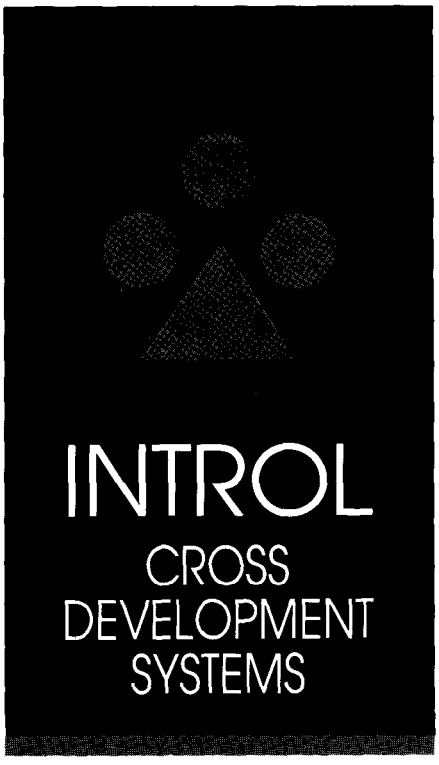
;--- BS (backspace)
             CJNE      A,#BS,L?notbs ; backspace
             POP       ACC             ; discard char
             GetXData  CurrentCol      ; tick column
             DEC       A
             JNB       ACC.7,L?bs1     ; but stick at zero
             CLR       A
L?bsl        MOV       B,A             ; to new column
             GetXData  CurrentRow      ;   of current row
             PUSH      B
             PUSH      ACC
             CALL      LCDSetCursor    ; set cursor there
             MOV       A,#BLANK        ; write a blank
             CALL      LCDSendChar
             POP       ACC
             POP       B
             CALL      LCDSetCursor ; set cursor again
             SJMP      L?done

L?notbs      SJMP      L?print         ; print all other chars!
L?setcurs    CALL      LCDSetCursor ; common point for
                                       ;   cursor setting
             SJMP      L?done

<<< code for printable characters omitted >>>

L?done       RET
LCDSendChar  ENDPROC
```

listing 3—*Control characters are* ***handledbythe*** *LCD in a manner similar* ***to thatofa*** *video* ***display terminal.***



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

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

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

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

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



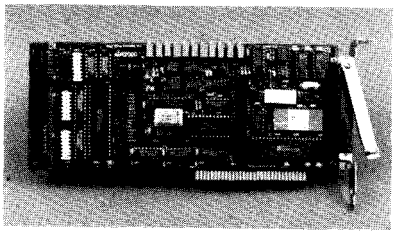
INTROL CORPORATION

9220 W. Howard Ave.. Milwaukee, WI 53220
414/327-7171 FAX: 414/327-7734
Quality Software Since 1979

Reader Service #134

PC Bus Data Acquisition and Control Cards

Quality U.S.-manufactured cards for single user, OEM, or embedded applications.



ADA2000 - $589

- 8-Channel, differential, 12-bit, 20µs A/D
- Programmable gains of 2, 4, 8, & 16
- Three 5-MHz timer/counters
- Two 12-bit D/A outputs
- 40 Digital I/O lines
- 120 signals through a single slot!

Real Time Devices, Inc. designs and manufactures a broad line of cost-effective industrial/scientific interface cards for the PC/XT/AT bus. Our commitment is to offer only high-quality U.S.-designed and manufactured interfaces with emphasis on signal quality and ease of use for OEM applications. All our cards are backed by a one-year warranty, and 30-day NO-RISK return policy. Call today to request your free catalog and discuss your application requirement1

AD1000 8-channel 12-bit 20 ps A/D; sample & hold; three 5-MHz timer/counters; 24 TTL digital I/O lines. $325
AD2000 S-channel differential 20 ps A/D; sample & hold; three 5-MHz timer/counters; 2,4,8,16 prog. gain; 16 digital I/O . . $495
AD100 l-channel single-ended 12-bit integrating A/D; 1,10,100 prog. gain $159
AD200 4-channel 12-bit 125 µs A/D; three 5-MHz timer/counters; resistor-configurable gains; 24 digital I/O lines..... $259
AD500 I-channel 12-bit integrating A/D; programmable gains of 1,10, & 100. Extremely stable. accurate & sensitive. $259
ADA100 Single-channel, differential input, 12-bit integrating A/D; 8-bit D/A output; gains of 1, 10, & 100. Plus 10 digital I/O lines. $215
ADA300 I-channel 8-bit 25 ps A/D; single 8-bit D/A; 24 TTL digital I/O lines $259
DA600 Fast settling dual/quad 12-bit D/A, internal double buffering $207/298
DG24/96 24/48/72/96-line TTL compatible digital I/O cards; NMOS 8255-based. Opt. buffers and pull-up resistors $1101274
TC24 Five 5-MHz timer/counters; uses powerful AM9513 chip; 24 digital I/O lines from NMOS 8255 PPI chip $218
ATLANTIS High-performance data acquisition software; foreground/background operation; maximum 25-KHz rate; supports hard disk streaming; pull-down windows.. $250

Custom/OEM designs on request!

Real Time Devices, Inc.
820 N. University Drive
P.O. Box 906
State College, PA 16804
Phone: 814/234-8087
FAX: 814/234-6864

Reader Service #156

The LCDTEST program, however, is both a complete check of the hardware and a sample application for the LCD driver code. It copies characters from the serial input (9600 bps, 8N1) to the LCD driver and the serial output; you can examine the driver by just typing characters on your PC's keyboard.

Al though the driver code appears to be a "dumb terminal" because you can't position the cursor from your spin in a wait loop.

The HD44780's Select, Read/Write, and Enable inputs are driven by latches rather than directly from output bits. You must perform reads and writes to specific addresses with specific data bits to control these latches, as summarized for BASIC-52 in Figure 4.

Listing 1 shows the assembly language code required to write a single byte to the LCD controller. As mentioned above, the I/O addresses are permanently stored in EPROM at locations CmdAddr, DatAddr, and StrobeAddr. The GetCWord macro fetches those address into DPTR (Data PoinTer Register) in preparation for a MOVX to read or write the byte.

Assembly language programs are faster than BASIC-52 code, so there must be a way to throttle the program down to ensure that the HD44780 is always ready for the next data byte or instruction. Although I've used delay loops before, it seemed a shame to waste all the input hardware on the RTC-LCD board, so this time I used a routine to check the BUSY status bit. The HD44780 turns BUSY on whenever it cannot accept new data, so the LCD driver simply waits for BUSY to go off before sending the next byte.

Listing 2 shows what's required. The routine in the Furnace Firmware code (and available on the BBS) is somewhat more complex, because it includes code to prevent a permanent hang if the BUSY bit is stuck active, which might happen if the LCD panel is disconnected. Although the code reads a full byte from the HD44780, only bit 7 has any significance.

Listing 3 handles the (few!) control characters needed for a dumb terminal interface. Notice how much code is required for a simple backspace character! All other characters are displayable, so you can access nearly all of the HD44780's internal character set.

The LCD driver has several options that may adapt it to your application, each controlled by a bit in the LCDControls variable. The two most useful are LCD BLANKING, which determines whether the display is blanked during upward scrolls, and LCD_CRLF, which forces a linefeed after each carriage return character. You can also suppress scrolling and force wrapping instead of a CR/LF at the end of each line.

LCDTEST uses very simple polled serial port handlers, so if you try sending a file to the LCD (as I did!), you will find some missing characters. It takes about 100 ms to scroll a 4x20 LCD, so a 9600-bps data stream will flush about 100 characters down the drain during each scroll. The Furnace Firmware code will use heavy-duty, interrupt-driven, double-buffered serial handlers, so that problem will simply go away. ***[Editor's*** *Note: Software for this article is available on the Circuit Cellar BBS or on Software On Disk #15. For downloading and ordering information,* *see page 77.1*

STAY TUNED

OK, now we have a CPU and a display. Next, the keyboard!

If you have any questions or suggestions about this ongoing project, the best way to get in touch with me is through the Circuit Cellar BBS. Because the firmware will certainly change as I build this project, that's also the best way to get the latest versions of the code.

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

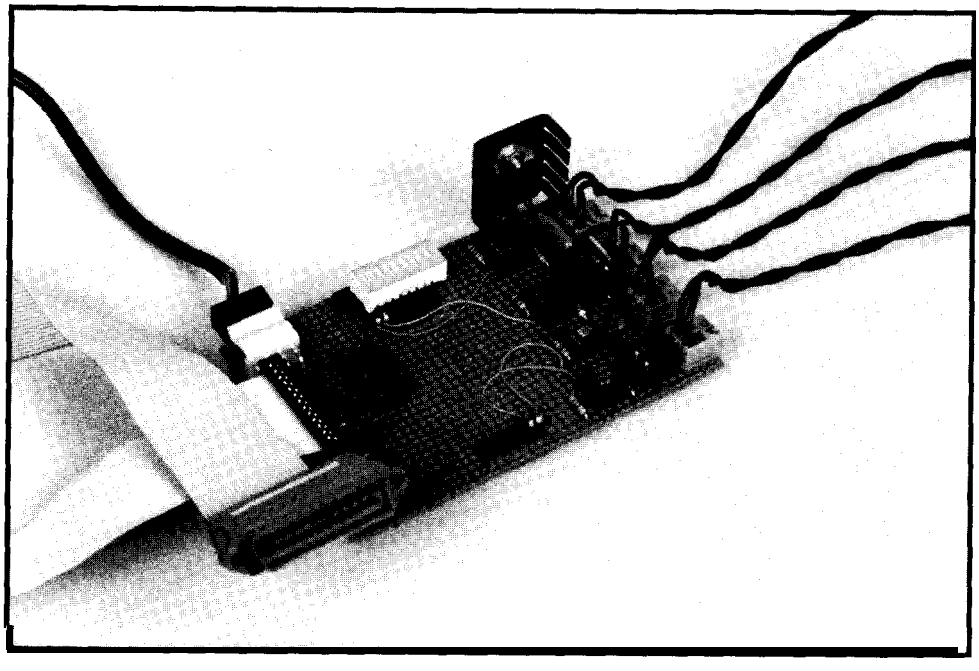
2 16 Very Useful
2 17 Moderately Useful
218 Not Useful

FROM THE BENCH

Jeff Bachiochi

Power Control Basics

Choosing the Best Digital Power Control Option for your Application



Ever had one of those days? You can usually tell early on: Things begin going wrong right away, although innocently at first. On any day but a holiday I am not the first one to show signs of life in the morning. However, that morning there was no school and I was alone in the kitchen foraging for food. Bagels, forced into narrow toasterslots, sent smoke signals as panic-stricken sleepers arose all too quickly to the screams of a wailing smoke alarm. Opening windows replaced the smoke-filled air with clean, 8°F air.

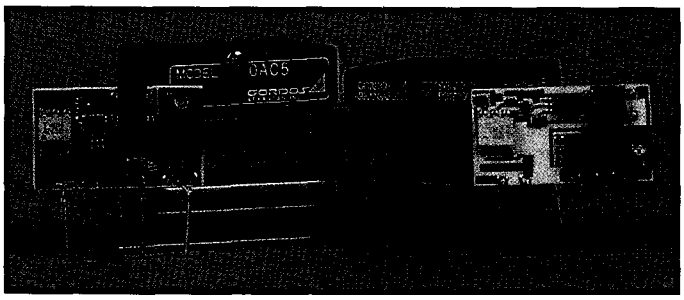
"Now the furnace will have to run awhile to warm... wait, I don't hear anything," I thought. "Why isn't it running? The thermostat says...hmm, there's no display."

I happen to have one of those LCD setback thermostats. Luckily, the problem was old nicads. Unluckily, I had no spares.

By the time I had the house in order and got to work, it was... well, let's just say Tracey was already collecting money for the "pizza-for-lunch" run. Both the regular and the decaf pots were empty so I had to settle for a Coke. My coins made a tinkling sort of sound as they fell out the coin return. Looking the coins over I expected to find that they were Canadian. No, the machine was just being temperamental. I tried them again, this time with a well-placed thump to the right side of the machine with my fist to be persuasive.

Plunk, out fell my Coke. It didn't stop at the little can catcher outlet. Instead, it popped out onto the floor. I could havejust waitedafewminutesbeforeopeningit if ithadn't fallen onto the sharp corner of the machine. Pfssst. Soda was squirting over everything as it rolled away from me. I grabbed a trash can to throw over it. At least I could confine the area of destruction. Unfortunately, the barrel was half full of trash and I ended up creating quite a sticky mess. Luckily, no one was in the hall and I scrambled to pick up the mess as quickly and quietly as possible. No one would ever know this happened, except maybe Mary Ann the cleaning lady.

I settled for a stop at the water fountain on the way back to my desk. I had to get something accomplished today. Writing technical manuals is not my favorite part of product development, but new products can't live without them. Besides, I couldn't get into any more trouble if I was sitting behind my own desk.



While the word processor was bringing up my latest file, I flipped through my CDs to find some appropriate background music. I won't go into the type of music I listen to because it changes so often. But, I will say it is not, nor has it ever been, opera! Out of respect for others in the office, I use lightweight headphones plugged into a portable CD player on my desk. With these I can still hear the phone if it rings.

"Seems as though I've been working on this manual forever. This afternoon's work should wrap it up," I reflected. Ring-ring. "Technical help on line 11."

"Thanks, Joan," I reply while slipping the headphones off my ears and down around my neck. "Hello, may I help you?" The caller asked if I had a part number for LED arrays. The info was just out of my reach. As I stood and reached for the manual, I heard a CRASH and got this tugging at my neck. There is not much more to add here. On my hands and knees I scooped up what I could, putting the pieces into a box. Someday I'll go back and look at it. Right now I wished I had left it on the floor. If I had, I might never have knocked out the power plug to my computer. Losing the afternoon's work seemed inevitable.

On the way out of the office I picked up my mail. Seminar advertisements, subscription renewals, and oh yes.. .two new CDs from the record club. Some things are beyond our control.

THE CONTROL OF INANIMATE OBJECTS

Fortunately, some things are within our control. We are the masters of our habitat controlling temperature, illumination, and sound whenever we adjust the thermostat, turn on a light, or tweak the stereo. The switch is the key ingredient to our control, providing two functions. First, it actually makes and breaks the connection between the power source and the appliance. In addition, some kind of insulation, usually plastic, creates an isolation barrier between the dangerous currents and our bodies.

OPEN LOOP VS. CLOSED LOOP

Have you ever wondered if the refrigerator light goes off when you close the door? This is an example of open-loop switching. There is a switch on the door frame that is supposed to turn off the light, but how do we know when the switch fails? Have you checked lately? If you have, then you have demonstrated how to close the loop, in this case by pressing the switch with the door open and using visual feedback to tell whether the light goes off.

When we turn up the thermostat in our home, does it actually get warmer? If the furnace starts running, that's feedback telling us it should be getting warmer. If we believe the thermometer, our eyes are providing the feedback, which says it seems to be **warming** up. But most of us are not convinced until we actually feel warmer. Three senses (sensors) each give varying amounts of feedback which closes the loop by verifying proper operation.

THE INS AND OUTS

The typical home computer can't tell how warm it is or how to turn on the reading lamp, but it does have input/output capabilities. Most of us are familiar with the paral-

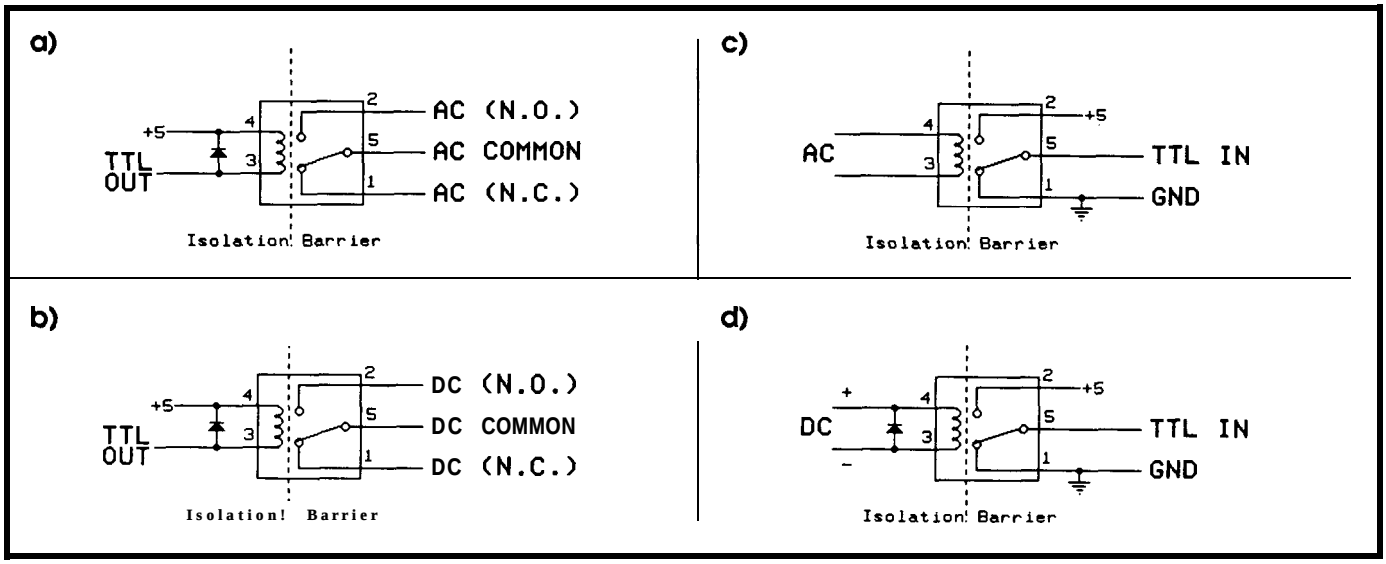


Figure 1 — *(a) An AC relay uses an open-collector output to control an isolated voltage. (b) A DC relay is essentially identical in concept to an AC relay. Both use the open-collector output from the TTL source to control an isolated 'real-world' voltage. (c) The same relay shown in (a) can be used as an AC sensing device. The relay requires no modification to switch from one function to the other. (d) Once again, the similarity of DC relays to AC relays is shown. The DC relay used for control in (b) becomes a DC sensing device when pin assignments are changed.*

lel printer port as an output device. Take a closer look. It is actually bidirectional, with a full eight bits out to send data to the printer, and four bits into the computer which indicate printer status like "not ready" or "no paper." The eight bits out of the computer are written by the processor to the printer port data register and are considered controlling bits. The four bits into the computer can be read by the processor at the printer port status register and are regarded as feedback.

IN THE BEGINNING...THE MECHANICAL RELAY

AC and DC voltages come in all shapes and sizes. The **sensing and** control of these require one of four basic circuits: AC control, DC control, AC sensing, and DC sensing. Each circuit performs two functions. The first function, level conversion, adapts computer control levels, normally 0-5-volt TTL, with the AC/DC levels associated with real-world equipment. The second function, isolation, protects the computer by electrically disconnecting it from potentially harmful voltages and currents found in the real world.

The mechanical **relay** accomplishes level conversion and isolation by using electromagnetically operated switches. A coil of wire energized by an applied voltage creates a magnetic field as current flows through the coil. This magnetic field pulls a movable contact-the switch wiper-from the normally closed contact in the deenergized position, to the normally open contact in the energized position. Since the switch is physically isolated from the coil, the switch contacts can be used to control an entirely different circuit. One coil can be used to move many sets of contacts, all completely isolated from one another.

Relay coils come in a wide variety of operating voltages. Typical DC coils are available from about 1 volt to over 100 volts. Turn-on time averages about 5 ms, while turn-off times are about 50% faster. Standard AC coils are available from 6 volts to well over 200 volts with switching times running a bit slower than their DC counterparts. Relay contacts can switch currents from 10 μA to over 30 A depending on the relay type. Contact life expectancy runs 50,000 to six million electrical operations, whereas mechanical life is about 100 times the contact life.

The output bits on the parallel printer port can sink about 20 mA. Even though some small reed relays can operate with only 10 mA, there are some reasons for not directly driving a relay. A logic low is necessary to turn this circuit on. The logic here is opposite from what one might expect. Adding **an** open-collector inverter will correct this, plus provide sufficient current to drive the relay. Writing a "one" to the port bit will energize the coil and writing a "zero" will deenergize it. The relay contacts can be used as the switch to control either AC or DC. Figures la and lb show how an open-collector output can be used to control an isolated voltage.

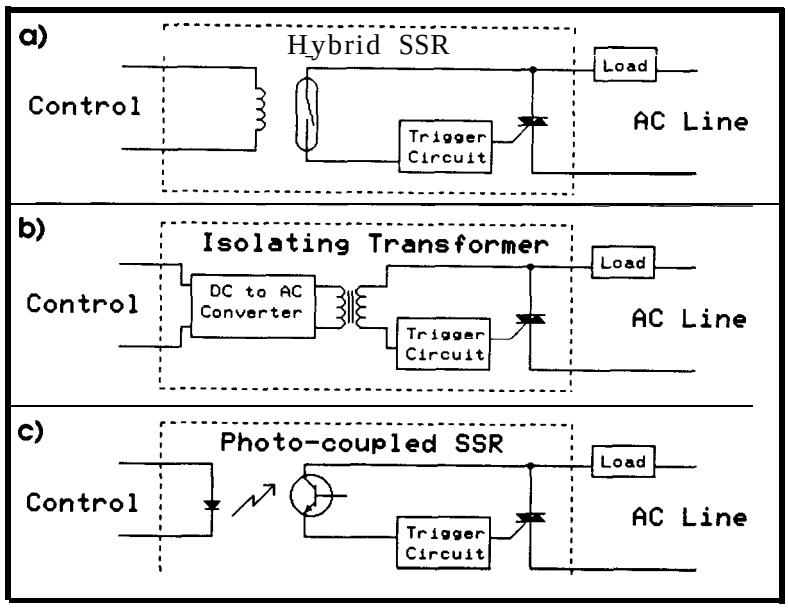


Figure 2—*(a) The hybrid SSR uses a reed relay for isolation. The hybrid is favored for switching large voltages and currents. (b) The transformer SSR creates a high-frequency signal on the primary of the transformer which is passed to a receiver on the isolated secondary winding. When a signal is received, a solid-state device is enabled. The transformer SSR contains no switch contacts to fail and no LED to degrade but is more expensive than either hybrid or optocoupled SSRs. (cl The optocoupled SSR uses an LED/photodetector pair for isolation. The LED's illumination is picked up by the photodetector which controls a solid-state device. The optocoupled SSR has the advantage of micro-second switching times.*

To sense the presence of voltage, select a relay which has a coil voltage and type (AC/DC) equal to that which you wish to monitor. Place the coil across the device so that it will energize whenever the device is on. The relay contacts can be used to switch the computer's input bits from a logic low to a logic high level whenever the relay is energized. Reading the parallel printer status port will reflect the status of the sensing relay and provide feedback to thecomputer. Figureslc and 1d show how a relay could be used to sense the presence or absence of an isolated AC or DC voltage.

Relaysdo have disadvantages. Typically, coil currents are high and contact switching times slow. Life expectancy is limited and switching states produces audible noise.

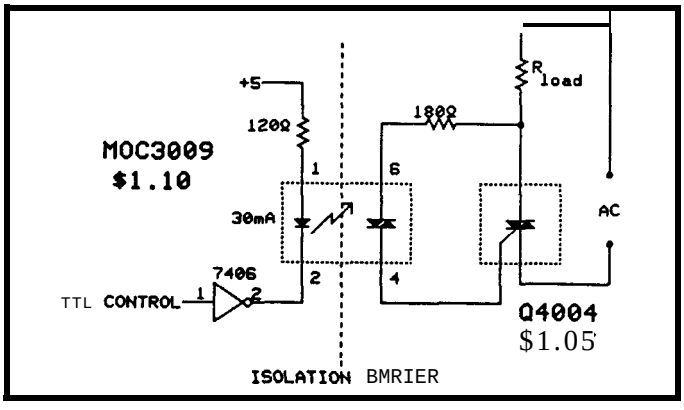


Figure 3-A *random turn-on, zero turn-off AC control device. Triacs control both ha/f cycles, like two SCRs in a parallel but opposite configuration.*

Contact switching also produces EM1 in the RF range. Shock or vibration can cause the contacts to bounce, making or breaking the circuit.

SOLID-STATE RELAYS

When compared to electromechanical relays which have been around for 90 years, the solid-state relay,brSSR,isarelatively **new component,** yet it is widely accepted as a significantly superior device in many respects. SSRs have a longer operating life, yielding a lower overall cost even though they are initially more expensive. SSRs are faster and have no bounce, arcing, or chattering problems associated with mechanical contacts.

Input-to-output isolation can be handled a few different ways in SSRs. Figure 2 illustrates three approaches.

The hybrid SSR uses a reed relay for isolation. The relay's contacts are used to control a solid-state device capable of switching large voltages and currents.

The transformer SSR creates a high-frequency (50–500 kHz) signal on the primary of the transformer. The signal is passed to a receiver on the isolated secondary winding. When a signal is received,a solid-state device is enabled as in the hybrid SSR.

The optocoupled SSR uses an LED/photodetector pair for isolation. The LED's illumination is sensed by the photodetector and controls a solid-state device as above.

The transformer SSR has some advantages over the hybrid and optocoupled SSRs. It contains no switch contacts to fail as in the hybrid, and no LED to degrade as in the optocoupled SSR, but it is more expensive.

The optocoupled SSR has the advantage of fast microsecond switching times and, although zero-crossing voltage turn-on is standard, random and peak voltage turn-on is available from several different manufacturers.

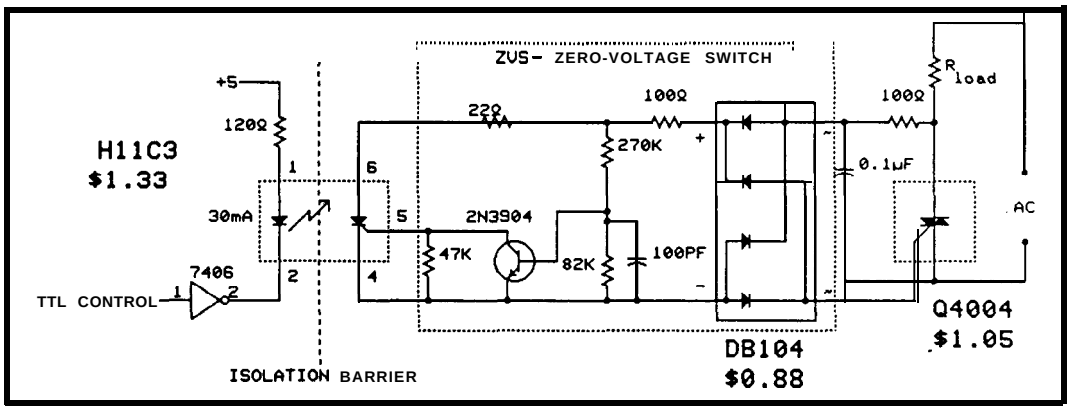
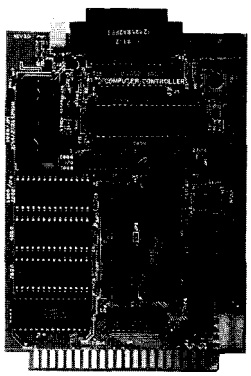


Figure 4-A **zero-crossing defector has been** *added to* **the** *triac* **control.** *By* **adding a** *zero-crossing* **detector,** *SCR/triac* **gates can be** *syncronized for zero* **or peak turn-on.**

BCC52
BASIC -52
COMPUTER/CONTROLLER



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

The BCC52 contains sockets for up to 48K bytes of RAM/EPROM, an "intelligent" 2764/128 EPROM programmer, three parallel ports, a serial terminal port with auto baud rate selection, a serial printer port, and is bus-compatible with the fug line of BCC-bus expansion boards. BASIC-523 lull floating-point BASK: is fast and efficient enough for the most complicated tasks, while its cost-effective design allows it to be considered for many new areas of implementation. It can be used both for devebpment and end-use applications.

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

MEMORY
- expandable to 62K bytes
- five on-board sockets
- up to four 6264 (8Kx8) static RAM
- either a 8K 2764 a 16K 27128 EPROM

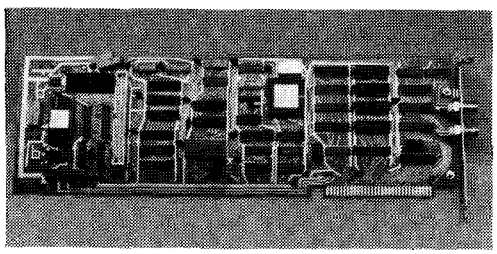
INPUT/OUTPUT
- console I/O RS-232 serial pat
- line printer RS-232-serialportt
- three 8-bit programmable TTL-compatible parallel I/O ports using a 8255 PPI
- alternate console RS-422/RS-485

To Order Call
1-800-635-3355
TEL: (203) 871-6170
FAX: (203) 872-2204
TELEX: 643331

		Single Qty.	100 Qty.
BCC52	BASIC-52 Controller Board with 8K RAM	$189.00	$149.00
BCC52C	Lower-power all-CMOS version of the BCC52	$199.00	$159.00
BCC52I	Full industrial temperature range	5294.00	$220.00
BCC52CX	CMOS, Expanded BCC52 w/32K RAM	$259.00	$159.00

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

Reader Service #145



PC BASED DIGITAL SIGNAL PROCESSING AND DATA ACQUISITION

- TMS320C10 based Model 10 - 25 Mhz
- TMS320C25 based Model 25 - 40 Mhz
- Optional 12 Bit, 110 Khz A/D and D/A
- Development Software, including Debugger
- Applications Software: FFTs, Spectral Analysis Program
- Optional Continuous, No Gap sampling to/ from disk at high rates
- Prices start at $650.00

DALANCO SPRY
89 Westland Avenue
Rochester, NY 14618
(716) 473-3610

Reader Service #118

ZERO-/PEAK-/RANDOM-CONTROLLED TURN-ON

SCRs and triacs are used as solid-state AC control devices. SCR conduction can be controlled on half of the AC wave cycle. An SCR will conduct when a voltage is applied to its gate input. Gate current enables conduction until the SCR's current has dropped (at the next zero crossing), even if the gate voltage is removed. For this reason, steady-state DC should not be controlled using SCRs or triacs because the current must cease in order for the device to turn off. Figure 3 is a random turn-on, zero turnoff AC control device. Triacs have control of both halves of the cycle, like two SCRs in a parallel but opposite configuration. By adding a zero-crossing detector, SCR/triac gates can be synchronized for zero turn-on or peak turn-on. All SCR and triac devices are zero turn-off. Figure 4 shows a zero-crossing detector added to the triac control.

In general, resistive, capacitive, and nonsaturating inductive loads should use zero turn-on SSRs. Currents through these loads are mostly in phase with the voltage. Turning on the load at minimum current will reduce the EMI/RFI normally generated at high *di/dt* values (fast change in current). However, the saturated core of an inductive load will cause current to lag behind voltage such that minimum current may occur at peak voltage. In this case peak turn-on will reduce EMI/RFI.

Zero-crossing detectors can be combined with optocouplers and triac drivers in one package. Motorola's MOC line does just that. The MOC3030 series, shown in Figure 5, gives 7500 volts of isolation through the optocoupler plus the **correct** gatingforan external triac to be turned on only at zero crossings of the AC line. An SCR's or triac's failure mode is normally shorted, so provisions must be made to limit load current. A current-limiting fuse can be used in series with the load for most small applications.

Several conditions exist which could hold the output device in a conducting or latched mode. Load currents which do not drop to less than about 300% of the gate-holding current will not stop conduction. Rapidly rising voltages (*dv/dt*) fromline transients can cause the inherent capacitance of an SCR/triac to support adequate current flow through the device to cause conduction. An R/C snubber, zener, or varistor network will reduce *dv/dt*.

Proper use of any device requires its application to fall below the ratings of the device. This is trueof SSRs as well. Alwaysanalyze and measure the complete load conditions before choosing an SSR. This will prevent premature degradation or failure. As with any other solid-state device, take other environmental factors into account, like temperature, when sizing an SSR.

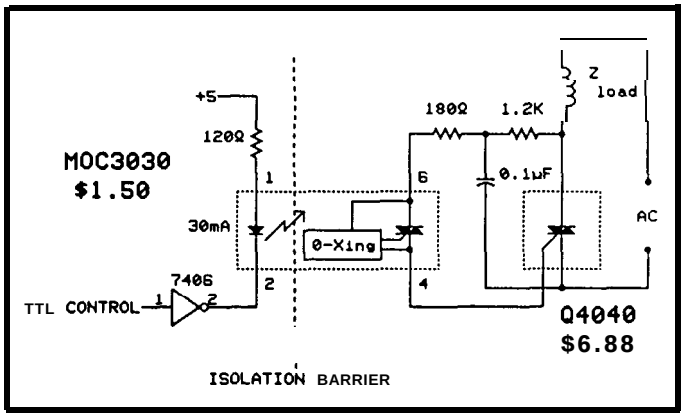


Figure *&-Motorola's MOC line combines zero-crossing detectors with optocouplers andtriac driversin one package. The MOC3030 series gives 7500 volts of isolation through the optocoupler.*

INDUSTRY-STANDARD I/O MODULES

As discussed earlier, four basic functions are necessary for the control and sensing of AC or DC. Each of these functions-AC control, DC control, AC sensing, and DC sensing-an be created using solid-state devices. When assembled (usually on a small circuit board), they can create a functional module with four or five leads. A standardized five-lead footprint has been accepted by industry for these modules. The case size is 1.7' by 0.6" by 1.25" and has a screw to secure the module to the equipment.

Standard modules are available with different output and input specifications. Shown in Photo 1 are the most widely used of the I/O modules. Figures 6a–6d are equivalent circuits of the four basic I/O module functions.

USING THE I/O MODULES

Place input modules across the load to sense when power is applied and output modules in series with the load to control power to the load. Input modules can also be used to monitor contact closures in such sensors as thermostats or limit and proximity switches. Notice that DC modules have a polarity associated with them.

[Editor's Note: Extremely dangerous and life-threatening voltages and currents arepresent on these devices. Proceed with caution. Isolate all circuitry which could

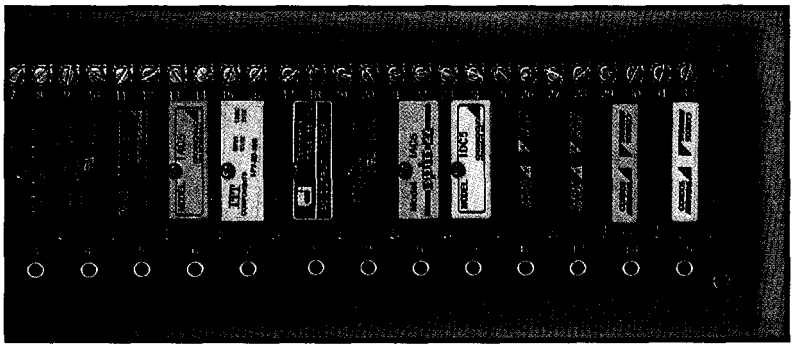


Photo 1 -*Optoelectronic parts are frequent/y color coded for function and value. Here, the following code applies:*

Type	Part #	Color	logic Connection	Real-World Connection
Input-AC	IAC5	yellow	5 VDC	90–140 VAC
Output-AC	0AC5	black	5VDc	12– 140 VAC
Input-DC	IDC5	white	5 VDC	10-32 VDC
Output-DC	ODc5	red	5VDc	5–60 VDC

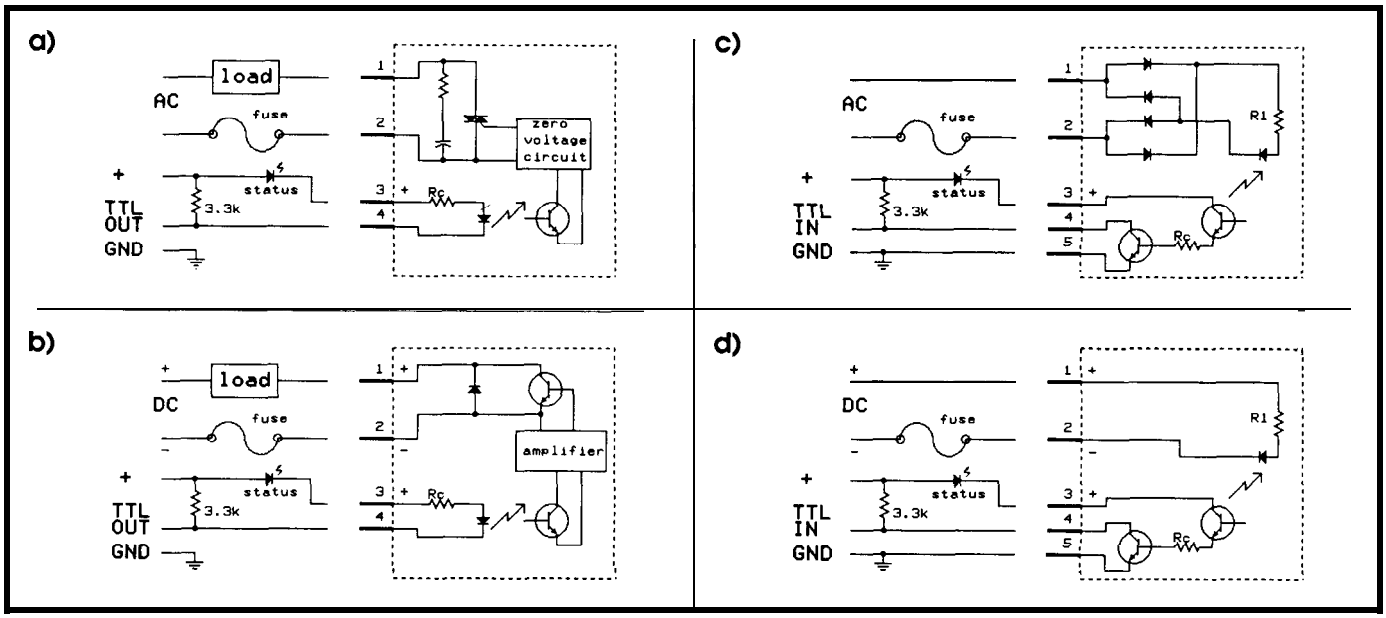


Figure 6—*(a) An AC Output Module. (b) A DC Output Module. Notice that there is a polarity associated with the module. (c) An AC Input Module. (d) A DC* **Input** *Module. Again, notice the* **polarity** *associated with the module.*

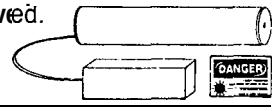
come in contact with ***anything*** *conductive,* ***including*** **your** *body. Assume every component is a potential* killer.]

The following example, interfacing to a printer port, allows some experimentation using a personal computer. As shown in Figure 7, an AC motor is controlled by bit DO of **the** printer's data output port using an OAC5 module equivalent. The motor's shaft has a magnet on it, and the **magnet** passing a magnetic or hall-effect switch provides feedback to D7 of the printer's input status port via an IDC5 module equivalent. In this case, the feedback indicates the shaft is turning; software could determine its speed in RPMs by counting pulses per minute. Next, we are going to solve the ageless refrigerator mystery. An 01X5 module equivalent is used to control a solenoid

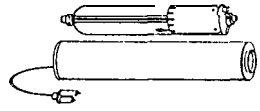
LASERS

MEREDITH INSTRUMENTS
P.O. Box 1724 I6403 N. 59th Ave.
Glendale, Arizona 85301
(602) 934-9387

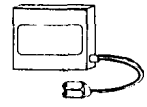
HELIUM NEON KITS consist of a He Ne Head and matching Power Supply. FDA approved.
- 1 mW Kit with 12 VDC Power Supply $110.00 with 110 VAC Power Supply $135.00
- 5 mW Kit with 12 VDC Power Supply $195.00 with 110 VAC Power Supply $220.00



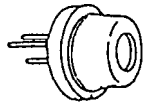
HELIUM NEON TUBES & HEADS **Hard Seal units.**
- .5 mW Tubes $35.00 • 3 mW Tubes $75.00 • 1-2 mW Heads $50.00
- 1mW Tubes $50.00 • 5 mW Tubes $100.00 • 5-7 mW Heads $125.00



HELIUM NEON LASER POWER SUPPLIES **High Voltage Switch-mode Regulated Factory made units.**
- 9 VDC input micro supply for .5 - 1 mW units only, .66"x1.06"x2.2" $75.00
- 12 VDC input adjustable output supply for I-7 mW units $75.00
- 1101220 VAC input adjustable output supply for 1-10 mW units $95.00



LASER DIODES & DRIVERS
- New visible 670 nm, 3 mW diodes $100.00 New L.D. Drivers $35.00
- New Collomated 800 nm, 3 mW diodes $50.00 Adjustable Collomators $25.00



LASER ACCESSORIES-optics, Holography **Kits, Scanners, Light Show Equipment, Books, Hardware and more.**



FREE CATALOG-call **or write** today for **our latest catalog or to place a C.O.D. order.**

Reader Service #141

CIRCUIT CELLAR INK ADVERTISER'S INDEX

Reader Service Number	Advertiser	Page Number
101	Andratech	11
102	A.T. Barrett Associates	78
104	Avocet	c4
105	Aware Electronics	78
106	Binary Technologies	48
107	Byte Boss Intelligent Systems	78
108	Cabbage Cases	11
109	CAD Software, Inc.	59
110	Catenary Systems	25
*	Ciarcia Design Works	79
111	Ciarcia's Circuit Cellar	48
112	Circuit Cellar	62
113	Circuit Cellar	62
114	Computer Doctors	23
115	Computerwise	71
116	Cottage Resources	50
117	Covox Inc.	31
118	Dalanco Spry	61
119	Davis Instruments (DIGITAR)	47
•	Deus Ex Machina Engineering	78
120	Dycor Industrial (DA/M)	39
121	Electronic Energy Control	53
122	Emerald Microware	4
123	Engineers Collaborative	73
124	Express Circuits	11
126	F&W Communications	51
127	Gott Electronics	7
128	Grammar Engine	27
129	GTEK	c2
130	Hazelwood	67
131	HiTech Equipment Corp.	52
132	Information Modes	79
133	Innotec	47
134	Introl Corp	55
103	Jameco Electronics	79
135	Kolod Research	73
136	Komputerwerk	78
137	Laboratory Microsystems	27
138	Link Computer Graphics	40
139	Louis E. Wheeler	79
140	L.S. Electronic	6
141	Meredith Instruments	64
142	Micro Dialects, Inc.	15
143	Micro Resources	38
144	Micromint	22
145	Micromint	61
125	Micromint	79
146	Ming Engineering (Electronics 1 2 3)	54
147	Needham Electronics	14
148	NOHAU Corp	31
149	Nuts & Volts	40
150	Paradigm Systems	14
151	Parallax, Inc.	43
152	PC Boards	71
170	Prairie Digital	78
153	PseudoCorp	17
154	Quinn-Curtis	13
155	R & D Electronic Surplus, Inc.	38
156	Real Time Devices	56
157	Royer Associates	78
158	Sierra Systems	c3
159	Silicon Alley	26
160	Sintec Co.	67
161	Softools	45
162	Synetic Systems	37
163	Timeline	29
•	Tinney	72
164	T.O.A.D. Inc.	15
165	Traxel Laboratories Inc.	54
166	Unkel Software, Inc.	35
167	URDA, Inc.	78
168	Witts Associates	78
169	Wytec Company	6

SPECIAL INSERT ADVERTISER'S INDEX

Reader Service Number	Advertiser	Page Number
175	All Electronics	s7
176	Alpha Products	SC2
177	Arcatron	S19
178	Berry	s17
179	B.G. Associates	S19
180	ECT, Inc.	S21
181	Heath	S9
182	Home Automation Associates	s7
183	Home Automation Labs	S11
184	Kelvin	s5
185	Micro Digital	S21
186	Micromint	S23
187	Merrimack Valley	S11
188	New Micros, Inc.	S26/27
189	Parks Associates	S10
190	Sunnyside Solar	S11
191	Vesta Technology	S30

HA JAR ASSOCIATES

NATIONAL ADVERTISING SALES REPRESENTATIVES

NORTHEAST
Lisa D'Ambrosia
Tel: (617) 769-8950
Fax: (617) 769-8982

SOUTHEAST
Christa Collins
Tel: (305) 966-3939
Fax: (305) **985-8457**

MID-ATLANTIC
Barbara Best
Tel: (201) 741-7744
Fax: (201) 741-6823

MIDWEST
Nanette Traetow
Tel: (708) 789-3080
Fax: (708) 789-3082

WEST COAST
Barbara Jones & Shelley Rainey
Tel: (714) 540-3554
Fax: (714) 540-7103

```
10 CLS
20 PPWRITE=&H378 : REM Address of my printer's
   output port
30 PPREAD=&H379 : REM Address of my printer's
   input status port
40 OLD=INP(PPREAD) : REM Dummy read
50 GOSUB 140 : REM Display info
60 I$=INKEY$ : REM Key check
70 IVALUE=INP(PPREAD) : REM Get new status
   input value
80 IF OLD<>IVALUE THEN OLD=IVALUE : GOSUB 140
   :REM If changed,update info
90 IF I$="" THEN 60 : REM if no key pressed,
   check again
100 IF I$="1" THEN OBIT0=ABS(OBIT0-1) : REM If
   '1' pressed, toggle output bit0 value
110 IF I$="2" THEN OBIT1=ABS(OBIT1-1): REM If
   '2' pressed, toggle output bit2 value
120 OUT PPWRITE,OBIT1*2^1+OBIT0: REM Build and
   output the output byte value
130 GOTO 50 : REM Loop back and check for
   another key
140 LOCATE 1,1 : REM Home cursor
150 PRINT "Hit '1' to turn on or off the motor"
160 PRINT "Hit '2' to turn on or off the
   solenoid"
170 PRINT
180 REM Outputting a 1 in a bit position turns
   opto modules on
190 REM Outputting a 0 in a bit position turns
   opto modules off
200 IF OBIT0=1 THEN PRINT "The motor is on"
   ELSE PRINT "The motor is off"
210 IF OBIT1=1 THEN PRINT "The solenoid is on"
   ELSE PRINT "The solenoid is off"
220 PRINT
230 REM A 1 in bit7 of the input byte means the
   input device is on
240 REM A 0 in bit7 of the input byte means the
   input device is off
250 REM Bit7 is opposite of the other bits
   because it's inverted by the printer port
   hardware
260 IF (OLD AND 128)=128 THEN PRINT "The hall-
   effect sensor is on" ELSE PRINT "The hall-
   effect sensor is off"
270 REM A 1 in bits4-6 of the input byte means
   the input device is off
280 REM A 0 in bits4-6 of the input byte means
   the input device is on
290 IF (OLD AND 64)=64 THEN PRINT "The
   refrigerator light is off" ELSE PRINT "The
   refrigerator light is on"
300 PRINT : PRINT
310 RETURN
```

listing 1 -*Control code for the circuit below shows how easy it is to interface to the outside world using optoisolated modules.*

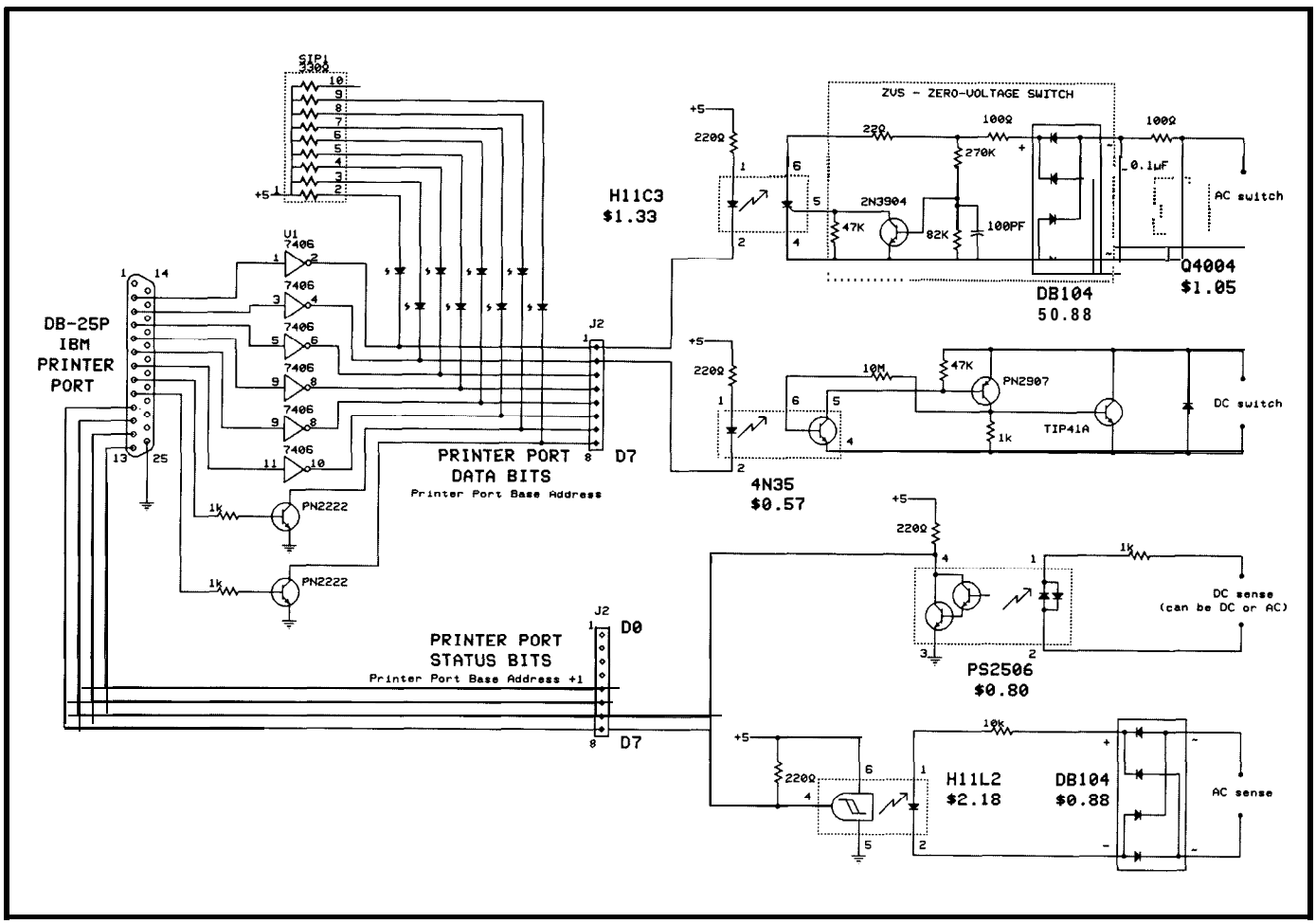


Figure *7-An AC motor is controlled by bif D0 of the computer's printer port using an OAC5 module equivalent. The motor's shaft* provides feedback to 07 of the computer's input status port via an *IDC5* module equivalent.

valve from data bit D1, which will close the refrigerator door switch. AnIAC5 module equivalent, connected across the light within the refrigerator, will indicate whether or not the power is actually turned off (see Listing 1).

I'm sure you'll find many uses for these I/O modules. All modules are priced around $10–$20 in small quantities. Many microcontrollers have modules available as peripheral I/O devices. Modulemanufacturers haverackmounts available to handle 8, 16, and 24 modules and are easily interfaced to a processor by tying them to a parallel port. For PC users, if more I/O control is necessary, special-purpose parallel I/O boards can be added to your system. New developments have allowed manufacturers to reduce the sizeof thesemodules to 0.4"in width. Others have added internal fuses and LED indicators to each module while retaining the original 0.6-inch width. High-power MOSFETs are being used on some DC output devices.

This is not a new industry, but one that is guaranteed to evolve along with other power semiconductors. Whatever direction you choose for your designs, be sure to include feedback sensing to make certain you have the situation under your control.+

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 manufacturingfields. In his spare* time, **Jeff enjoys** *his* family, windsurfing, and pizza.

SSR MODULES

Gordos Corp.
1000 North Second St.
Rogers, AR 72756
(501) 636-5000

Potter & Brumfield
200 Richland Creek Dr.
Princeton, IN 47671
(812) 386-1000

Opto 22
15461 Springdale St.
Huntington Beach, CA 92649
(714) 891-5861

Continental Industries, Inc.
5456 E. McDowell Rd.
Mesa, AZ 85205
(602) 985-7800

OPTOELECTRONICS

GE/RCA/Intersil Semiconductors
10600 Ridgeview Ct.
Cupertino, CA 95014
(408) 996-5000

Motorola, Inc.
3102 North 56th St.
Phoenix, AZ 85018-6606
(602) 952-3000

General Instrument Corp.
Optoelectronics Division
3400 Hillview Ave.
Palo Alto, CA 94304
(415) 493-0400

Part prices quoted from:
Digi-Key Corp.
(800) 3444539
(218) 681-6674

IRS

2 19 Very Useful
220 Moderately Useful
221 Not Useful

68000 K-System **Factory Direct Prices**

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

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

Boards currently in production:		AVAILABLE/N KIT FORM
K-BUS	12 Slots .8" centers, PC type power connectors	$129.95
K-CPU-68K	10MHz 68000 CPU, 2 ROM sockets (12 or 16MHz)	$129.95
K-MEM	256K static RAM or 27256 type EPROMs (OK installed)	$ 59.95
K-ICI	2 serial ports with full modem contmk (68681)	$ 99.95
K-FDC	Floppy disk 1/4 inch controller (up to 4 drives)	$ 99.95
K-SCSI	Full SCSI implementation using 5380 chip	$ 99.95
K-DMA	2 channel DMA controller using 68440 chip	$129.95
K-PROTO	General purpose/wirewrap board	$ 39.95
K-xxx-BE	Bare board with documentation for above	$ 39.95

Software:

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

Inquire about our UniQuad line of 68xxx Single Board Computers.
Quantity and package discounts available

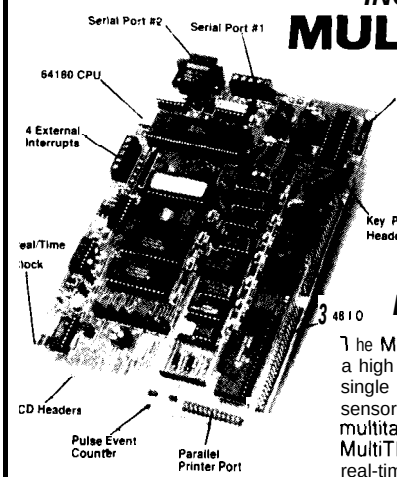
Terms: Check, Money Order, Visa, MasterCard—Prices include UPS ground shipment in continental US.

Hazelwood Computer Systems

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

Reader Service #130

"INCREDIBLE"
MULTI-TRAX™



NEW
Multitasking Programmable Controller
Features 100% Compiled Basic Interpreter

The MultiTRAX Microcontroller is a high capability cost-effective single board control and data! sensor acquisition system. With its multitasking BASIC compiler, the MultiTRAX Controller provides real-time response and extraordinarily fast program execution in a quick and effective programming environment complete with on-board EPROM programmer. With its expandable single board form factor and extensive I/O capability, the MultiTRAX Controller adapts to a wide variety of applications as a complete stand-alone system or as a comprehensive embedded controller.

- HIGH SPEED CMOS PROCESSOR
- 48 DIGITAL I/O LINES
- TWO RS-232 SERIAL PORTS
- MULTITASKING BASIC LANGUAGE COMPILER IN ROM
- ON-BOARD EPROM PROGRAMMER
- 96K BYTES OF APPLICATION MEMORY SPACE
- FOUR EXTERNAL INTERRUPTS
- PARALLEL PRINTER PORT
- SINGLE BOARD FORM FACTOR

EXPANSION MODULES: **Solder directly to single board computer**

- A/D Converter
- Real time calendar clock
- System Control ROM
- Keypad Module: 4 x 4 keypad with tactile feedback
- LCD Display module(s)
- Pulse/Event/ Frequency Counter

BASIC KIT STARTS AT ONLY $269.50!

SINTEC COMPANY
28 8th St., Box 410
Frenchtown, NJ 08825
CALL TOLL FREE **800-526-5960** N. J. **201-996-4093**

Reader Service #160

SILICON UPDATE

Tom Cantrell

Chips for Artificial Intelligence

I've Seen The Future-and It Is Fuzzy

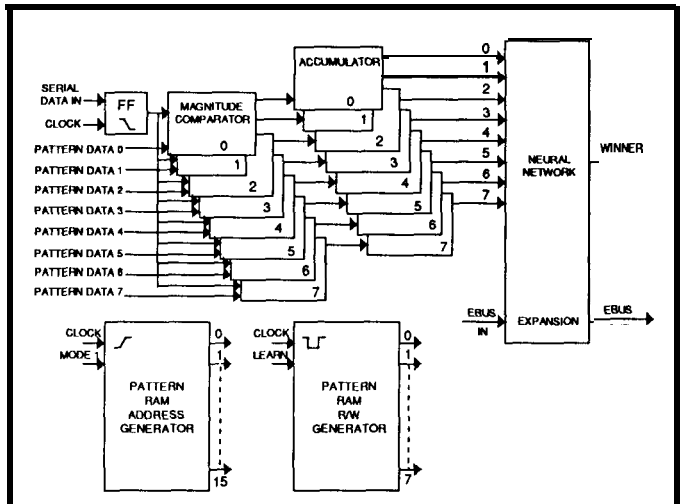
Let's face facts: The computer revolution has been a result of technology-the shrinking of tube-ridden behemoths to tiny chips-not architecture, where yesterday's concepts (i.e., digital stored program) persist with little more than cosmetic changes. The pace of change has never been great and, if anything, is slowing down. Indeed the term "architectural innovation" borders on an oxymoron these days. Is there a technology on the horizon that can break the chain of mediocrity, or will the '90s see more MIPS chasing fewer sockets? Fortunately, even the most jaded observer can see hope in the tea leaves.

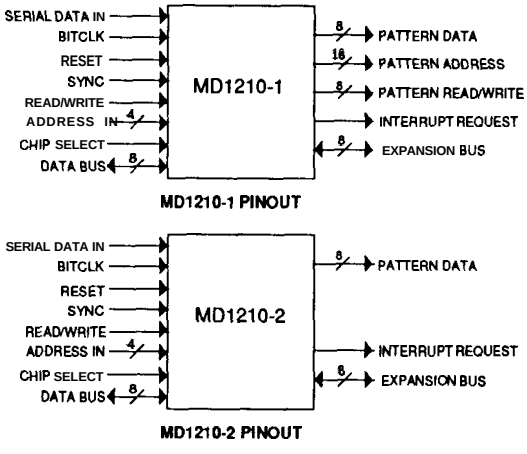
Until now, artificial intelligence has been more sizzle than steak, but finally some real neat (i.e., both neat and real) products are emerging that offer the potential for revolutionary improvements in computer capability. Neural Networks and Fuzzy Logic are ways to elevate machines to a higher level of consciousness, allowing them to "think" instead of just "following orders."

THE MD1210 FUZZY SET COMPARATOR (FSC)

The FSC isn't the "wonderchip" that will single-handedly turn your PC into R2D2. However, when it comes to recognizing sound, images, and other "fuzzy" (i.e., real-world, noisy) data, the FSC plays a crucial role.

The operation of the FSC is actually easy to describe, and that's part of the reason I like it. As the name implies, the chip'spurposeis to compare fuzzy sets with thegoal of determining the closest "match." The FSC has nine pattern inputs which can be configured to compare one unknown pattern with eight known (i.e., "learned") patterns (or vice-versa). Examining the FSC hardware, shown in Figure 1, shows that the unknown pattern is clocked in serially and simultaneously compared with eight known patterns stored in external RAM. During comparison, differences between each set are accumulated as illustrated in Figure 2. After pattern comparison completes, a back-end neural network determines the winner: The accumulator





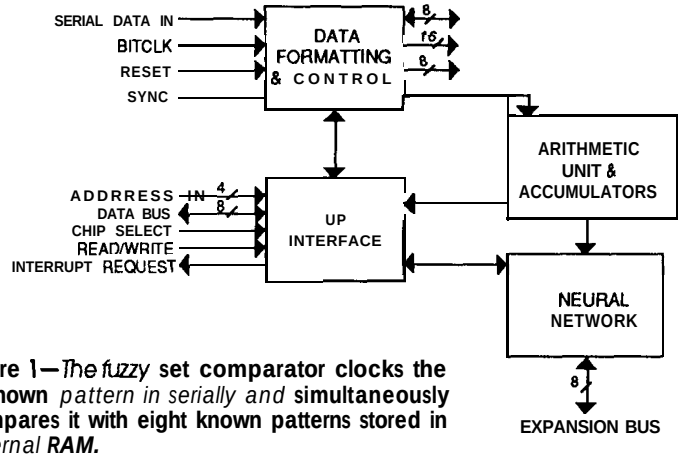


Figure 1—*The fuzzy* **set comparator clocks the unknown** *pattern in serially and* **simultaneously compares it with eight known patterns stored in** *external* ***RAM.***

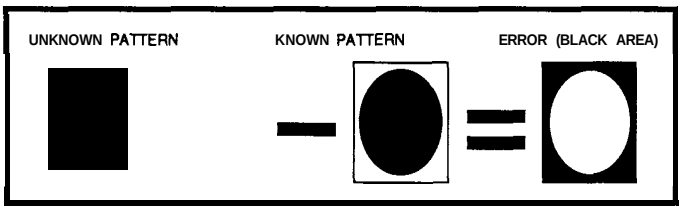


Figure 2-During *a* **comparison, differences** between **each set** *are* **accumulated and a back-end** *neural* **net determines the winner.**

with the lowest error corresponds to the pattern with the closest match.

Looking a little further beneath the surface makes the details of operation clearer. First, a pattern is characterized by its number of fields and number of bits per field. The product of the two determines the total bits per pattern. The relevant FSC specs are.. .

*Field Length-up to 64K fields
*Bits per Field-l-8
•Total Bits per Pattern-up to 64K bits

For example, a digitized image with resolution of 256 x 256 x 1 or x 2 can be directly handled by the FSC. If necessary, it's possible to overcome the 64K-bit pattern length restriction with a little external TTL which accesses multiple banks of pattern memory as shown in Figure 3.

As each field in the known and unknown pattern is compared, the absolute value of the difference between fields (i.e., the error) is added to the running sum in the **error** accumulator associated with each of theeight pattern memory inputs. This leads to a somewhat nonobvious restriction on the number of bits per field related to the fact that the error accumulators are only 16 bits long. For fields of few bits, each comparison can add only a small number to the accumulator. Of course, more bits per field can generate large errors per comparison. The thing to watch out forisaccumulatoroverflow.Forafieldlengthof 1 bit, overflow can't occur since the maximum pattern length, and thus potential accumulated error, are both 64K. At the other extreme, 8-bit fields could conceivably generate an error of 256 per comparison. In the worst case, the accumulator could overflow after only 256 (256 x 256 = 64K) comparisons. This has implications for applications that I will elaborate on shortly.

Now that the patterns have been compared and errors accumulated, the next step is to subject the measured differences to a threshold check. Patterns whose accumulators contain **a** value larger than that specified in **a** threshold register are immediately marked as losers. This programmable threshold register is a key to the "fuzzy" aspect of the chip. Indeed, setting the threshold register to 0 makes the FSC look for exact matches, eliminating its fuzziness altogether. The programmable threshold is what allows the designer to make application tradeoffs depending on how "different" the various patterns are likely to be, the costs of false negatives (fails to recognize a known pattern) versus that of false positives (recognizes an unknown pattern), and so on.

The final step within the FSC itself is to choose the closest match (i.e., lowest accumulator) from those that

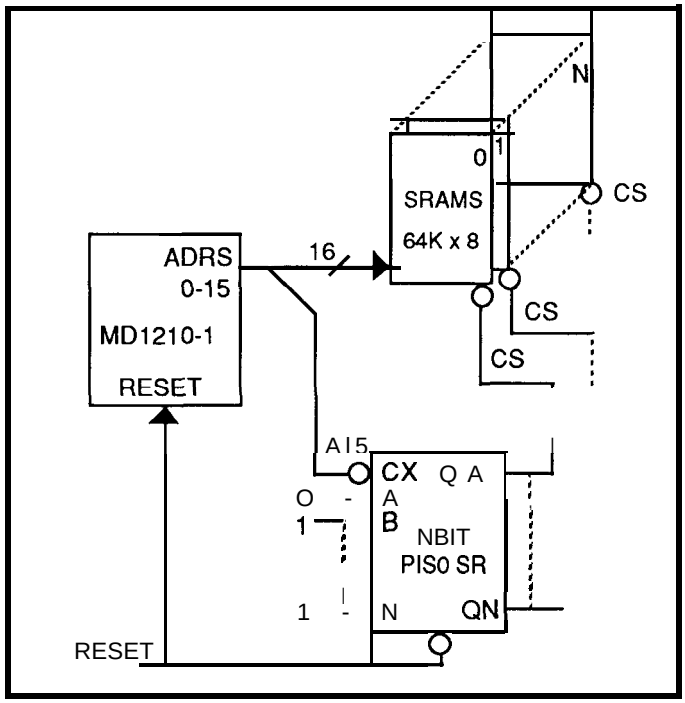


Figure 3—*It's* **possible to overcome** *64K-bit* **pattern** *length limitations by adding circuitry to access multiple banks of pattern memory.*

pass the threshold test. Here is where a neural network comes into play. The accumulator values are fed to a network of 128 neurons that has been trained to choose the minimum value.

But there's more. A second neural network is devoted, along with an expansion bus, to determining the overall winner from a conglomeration of up to 32 FSCs. Thus, the single FSC's eight-pattern capacity can be expanded straightforwardly to 256 patterns by simple replication of the single-device subsystem (Figure 4).

The process of choosing the single lowest error value from 256 entries (8 values per chip, 32 chips) is pretty easy; a small loop of code could take care of it. However, the neural network is much faster than a sequential scheme—only five clocks (two for each net plus one for arbitration).

THE **NEED FOR SPEED**

As you've seen, the way the FSC works is refreshingly simple. Indeed, the complete operation of the device is roughly described by a simple BASIC program shown in Listing 1. This example is set up to compare one unknown pattern with eight known patterns.

The patterns are all initialized with random data, but for illustration, the unknown pattern is synthesized from one of the known patterns. The **FUZZY: code** assigns variations of pattern 8 to the unknown pattern. The last option is chosen (i.e., left unREMarked) to generate a noisy version of pattern 8. After the desired value for the threshold is entered, the patterns are compared and errors accumulated. Finally, one winner is chosen: the known pattern which both most closely matches the unknown pattern and passes the error threshold test. Two runs (Figure 5)

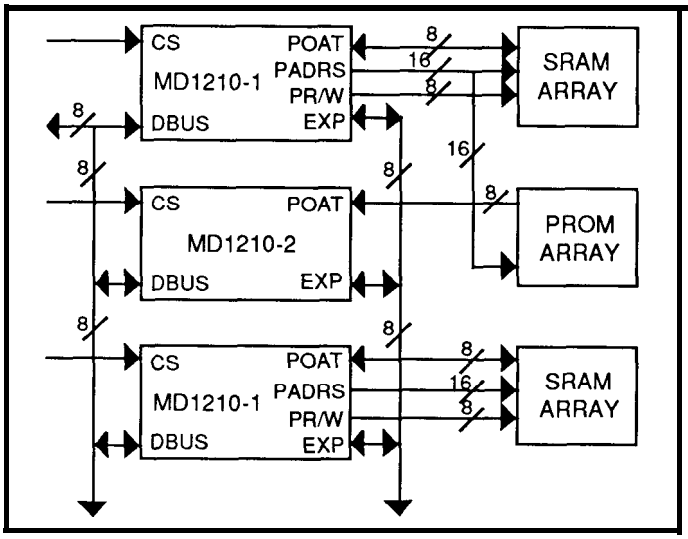


Figure 4—*The FSC's eight-pattern capacity can be expanded to 256 patterns by replication of single-device subsystems.*

show how "matchable" the noisy pattern is (note the much lower error for pattern 8 than patterns 1–7) and that the threshold must be set high enough to tolerate the expected noise level-otherwise a should-be winner will be rejected.

This simple example differs from the FSC in a couple of ways. First, the 16-bit BASIC INTS are twice the FSC's maximum 8 bits per field. To avoid the error accumulator problem mentioned earlier, the accumulators are made LONG INT (32-bit) and the value stored in the field is limited to 8 bits. Also, the final selection of a winner only simulates one (on-chip winner) of the two (on-chip winner, off-chip winner) FSC neural nets. Given that the purpose of this exercise was just to get a rough idea of the relative performance between a regular CPU and the FSC, you'll see why I didn't bother tweaking the BASIC program further.

I punched the code into my trusty Macintosh 11x (16-MHz MC68030 and MC68881). The relevant portion of the program, between the START and "END" messages, took about a second to execute. (This was compiled QuickBASIC code. Interpretive execution takes about 15 seconds.)

Go ahead, criticize my programming style, choice of language, complexity of instruction set, lack of megahertz, or whatever. Take your best shot-it won't be nearly good enough.

You see, the FSC can crank through the process at an astounding 20M bytes per second (8 pattern channels/bits x 20 MHz maximum clock frequency). Our example processes a grand total of 16K bytes (8 patterns x 2 bytes / field x 1024 fields). Thus, the FSC is over a thousand times faster than the Macintosh IIx!

Even if a regular CPU could get close, the FSC can easily up the ante. Remember, 32 FSC chips can work in parallel, boosting pattern bandwidth to 640M bytes per second. Of course, I could lash 32 Macs together to try to keep up, but ultimately it's an exercise in futility. In applications for which it was designed, the FSC is orders of magnitude faster than a regular microprocessor.

```
VARIABLES:
DEFLNG a    '/* error accumulators are 32-bit */
DEFINT b-z '/* others are 16-bit */
DIM known(8,1024) '/* known patterns */
DIM unknown(1024) '/* unknown pattern */
DIM accum(8)  '/* error accumulators */
RANDOMIZE TIMER

'/* threshold adjusts noise sensitivity */
INPUT "Enter threshold ";athreshold

LEARN: '/* train the FSC by loading patterns */
PRINT "Initializing patterns ";
'/* set up 8 learned ptrns, each 1024 fields */
FOR i=1 TO 8
  FOR j=1 TO 1024
    '/* pattern 1..8=small..large numbers */
    known(i,j)=INT(RND*&H20*i)
  NEXT j
PRINT "*";
NEXT i
PRINT

FUZZY: '/* synthesize an unknown pattern for
            compare by modifying pattern #8 */
FOR i=1 TO 1024
  '/* unfuzzy ptrn = exact match w/ptrn #8 */
  'unknown(i)=known(8,i)
  '/* "darker" pattern  */
  'unknown(i)=known(8,i)*0.9
  'IF i<128 THEN unknown(i)=INT(RND*&H100)
    ELSE unknown(i)=known(8,i) '/*framing err*/
  'unknown(i)=0  '/*1055 of signal */
  '/* random noise */
  unknown(i)=known(8,i)+INT((RND*&H40)-&H20)
NEXT i

COMPARE: '/* compare unknown pattern to 8 known
             patterns and accumulate errors */
PRINT "Starting comparison ";
FOR i=1 TO 8 '/* 8 known patterns to compare */
  FOR j=1 TO 1024 '/* pattern is 1024 fields */
    '/* accumulate error */
    accum(i)=accum(i)
             +ABS(unknown(j)-known(i,j))
  NEXT j
PRINT "*";
NEXT i
PRINT

WINNER: '/* search for winner (minimum error)
              and check against threshold */
amin=accum(1):win=1
FOR i=2 TO 8
  IF accum(i)<amin THEN win=i:amin=accum(i)
NEXT i
PRINT "Comparison end "

REPORT: '/* report results */
FOR i=1 TO 8
  PRINT "Pattern #";i;"Error =";accum(i)
NEXT i
PRINT "Threshold =";athreshold
IF accum(win)<athreshold
  THEN PRINT "Winner is pattern #";win
  ELSE PRINT "No winner"
INPUT i '/* hold display for viewing */
```

listing 1—*The complete operation of the FSC can be roughly described by a simple BASIC program.*

CHIP GOES TO SCHOOL

One difference between fuzzy and neural chips versus regular computers is fundamental: The former 'learn" while the latter are programmed. Thus, in an eerily human-like fashion, fuzzy devices can exhibit self-adaptive, goal-seeking behavior (i.e., a mind of their own).

A design based on the FSC is capable of different types of learning. On the fuzzy comparison side, learning occurs

when the pattern RAMs are loaded with known patterns. Another way of learning is provided by the programmability of the threshold register which can be changed to deal with varying noise levels in the patterns. However, a host CPU needs to take care of this since the FSC doesn't ever modify the threshold on its own.

Conceptually, another layer of learning should be possible by modifying the interconnection weights in the on-chip neural networks. For example, different FSCs might be assigned to recognize different types of data—say, audio and visual. The relative importance of each type of data in choosing a winner could be varied appropriately depending on the real-time circumstances at hand. However, the FSC's networks are hard wired such that each channel/device always has the same weight in the decision. On the other hand, the FSC can be directed to mask any or all patterns from comparison. Once again, as in programming the threshold register, a host CPU can offer a little tutoring, in this case by enabling/disabling pattern channels dynamically.

DOES IT REALLY WORK ?

The simple answer is yes...or no...or maybe. The ambiguity is fitting given the underpinnings of the FSC "beyond Boolean" technology.

Fortunately, you can get your hands on some iron and see for yourself. Micro Devices sells an FSC evaluation kit for an eminently reasonable $250. The kit includes a short-slot 8-bit ISA (PC) bus plug-in board, a disk with an FSC "front-panel" program, and various manuals and data sheets describing the board and FSC. The information is clear and complete (schematics, PAL equations, and so forth) and installation is a snap. The FSC board is designed to connect to an NTSC video source such as a VCR or video camera. The software included in the package allows you to load the pattern RAMs from the video source (or disk files) and monitor the FSC's operation as it goes through the process of comparing an unknown video pattern input to the learned patterns.

The idea is to capture various images (up to eight for a single FSC chip) into the pattern RAMs and then feed the FSC with new images to see just how well the chip can recognize them. The software provided gives you a window on all the FSC registers (Figure 6) which update automatically on the screen as the information is digested and analyzed. You can tweak the threshold and masking features as described earlier to see what effect they have on the result. For example, decreasing the allowable error threshold reduces false recognition of unknown objects, while increasing the threshold reduces sensitivity to noise, ambient light fluctuations, camera positioning changes,

$249. TERMINAL



Featuring
- Standard RS-232 Serial Asynchronous ASCII Communications
- 48 Character LCD Display (2 Lines of 24 each)
- 24 Key Membrane Keyboard with embossed graphics.
- Ten key numeric array plus 8 programmable function keys.
- Four-wire multidrop protocol mode.
- Keyboard selectable SET-UP features-baud rates, parity, etc.
- Size (5.625" W x 6.9" D x 1.75" H), Weight 1.25 lbs.
- 5 x 7 Dot Matrix font with underline cursor
- Displays 96 Character ASCII Set (upper and lower case)

Options-backlighting for display, R-422 I/O, 20 Ma current loop I/O,

COMPUTERWISE, INC.
302 N. Winchester . Dlathe, KS 66062 . 913-829-0600 . 800-255-3739

Reader Service #115

P-C-B ARTWORK MADE EASY !

Create Printed Circuit Artwork on your IBM or Compatible

* MENU DRIVEN
* HELP SCREENS
* ADVANCED FEATURES
* EXTREMELY USER FRIENDLY
* 1X and 2X PRINTER ARTWORK
* 1X HP LaserJet ARTWORK

* HP and HI PLOTTER DRIVER optional 49.00

REQUIREMENTS: IBM PC or Compatible, 384 K RAM DOS 3.0 or later.

PCBoards - layout program 99.00
PCRoute - auto-router 99.00
SuperCAD - schematic pgm. 99.00
DEMO - 10.00

Call or write for more information

PCBoards
2110 14th Ave. South, Birmingham, AL 35205
(205) 933-I 122

ader Service #152

```
ENTER THRESHOLD? 10000
INITIALIZING PATTERNS ********
STARTING COMPARISON ********
COMPARISON END
ERROR-PATTERN #1 = 115570
ERROR-PATTERN #2 = 104618
ERROR-PATTERN #3 = 93446
ERROR-PATTERN #4 = 86773
ERROR-PATTERN #5 = 84454
ERROR-PATTERN X6 = 82287
ERROR-PATTERN #7 = 84361
ERROR-PATTERN #8 = 16617
THRESHOLD = 10000
NO WINNER
```

```
ENTER THRESHOLD? 20000
INITIALIZING PATTERNS ********
STARTING COMPARISON ********
COMPARISON END
ERROR-PATTERN #1 = 118967
ERROR-PATTERN #2 = 107864
ERROR-PATTERN X3 = 97062
ERROR-PATTERN #4 = 91928
ERROR-PATTERN #5 = 83976
ERROR-PATTERN #6 = 82332
ERROR-PATTERN #7 = 84839
ERROR-PATTERN #8 = 16750
THRESHOLD = 20000
WINNER IS PATTERN #8
```

Figure 5—*Two sample* runs show how selection of the **threshold** *value influences the final* ***decision.***

and so on, which might otherwise cause a known object to be falsely unrecognized.

Experimenting with the kit sheds light on that "fuzzy" yes/no/maybe answer.

On the downside, the FSC is only a piece of the puzzle: It can't do anything without some degree of host CPU support and control. Indeed, for nontrivial applications, the CPU will need to perform significant preprocessing of the pattern data. Most basic is the need to "frame" the samples because comparison starts when a sync input to the FSC is asserted. This is straightforward for video applications (vertical sync does the job) but other applications will require a sync generator.

Don't forget the FSC simply does a brute-force comparison of the entire pattern sample-it won't "extract" a known feature buried within a large pattern. For the same reason, camera positioning is critical. The evaluation kit gives an excellent interactive demonstration of this. You can jiggle the camera and see the error increase dramatically. Automatic visual inspection systems—a likely candidate for FSC application-will require fairly precise positioning schemes (or more patterns representing different positions of each known object).

When choosing the basic pattern format, try to limit theinformation to that truly of interest. Besides speeding the comparison, this reduces the chance of error accumulator overflow. For video applications, the number of gray scales should be minimized (for many applications, black and white will suffice). Otherwise, the error accumulator can overflow as a result of slight lighting and positioning errors. Even with the evaluation kit's four-level (two bits per field) digitizer, the FSC is quite sensitive. A passing shadow is all it takes to throw the comparison off. The programmable threshold register allows compensating for a reasonabledegreeof real-worldvariability,butdon'texpect miracles. Try to structure your application (i.e., don't bite off more than the FSC can chew) and pay attention to minimizing extraneous information (and thus potential errors).

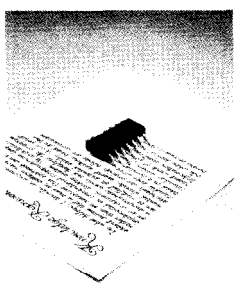
Can the FSC dramatically boost the intelligence of micro-based applications? The answer is an emphatic yes.

Limited Editions

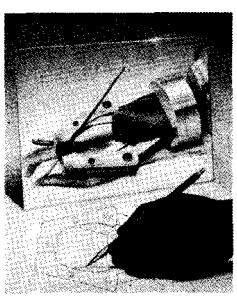
#G $55 #H $55



Programmable Hardware



Word Processor



Intelligent Reflections



Technological Breakthrough

#A $12 #C $70

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

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

The price of each print is shown at left. Order two or more and deduct 15%! Frames are only $39.50 each. For **shipping, add** $4 per order for unframed prints ($25 overseas); **for** framed prints, add $5 for one print and $3 **for each extra print** (ground). **No** frames shipped overseas. Full refund **if** no! satisfied. For VISA, MasterCard or AmEx orders, **call 1-318-826-3003.**

ORDER FORM cci

Qty	#	Title	Amount
-	-		
-	-		
-	-		
-	-		$

If you order two or more. deduct 15%. $
Framer 639.50 each) $
Shipping charges: See above $
Total $

☐ I have enclosed a check or money order to Robert Tinney Graphics. (Must be drawn on a U.S. Bank; no foreign collection, please.)
☐ Bill my 0 VISA ☐ MasterCard 0 American Express account:
Card No.: xpires:
Name:
Address:
City: State: Zip:
Country:

☐ Send a brochure showing your other prints.

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

"Neural" and "fuzzy" chips of the '90s will make even the whizziest microprocessor seem positively simple-minded. However, the micros won't disappear; they'll still have a role. After all, even a rocket scientist needs a calculator.

The MD1210 FSC is $38 in 100-piece quantities which makes the technology viable for volume commercial application. Micro Devices also offers other smart chips--the MD1212 Fuzzy Data Correlator (FDC) and the MD1220 Neural Bit Slice (NBS). If you want to get a head start on the future, give Micro Devices a call.

CONTACT

Micro Devices
5695B Beggs Road
Orlando, FL 32810-2603
(407) 299-02 11

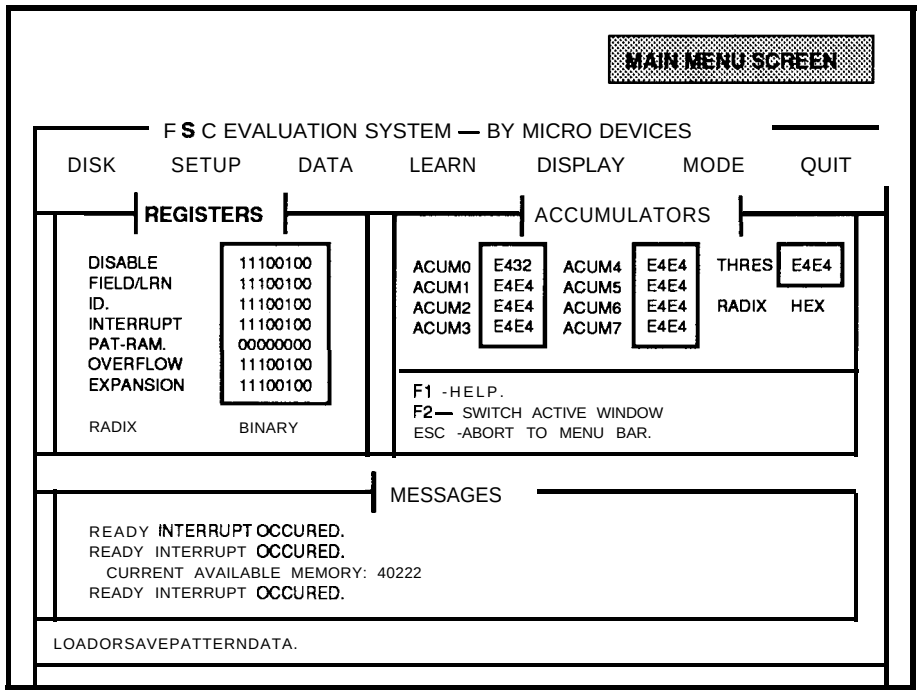
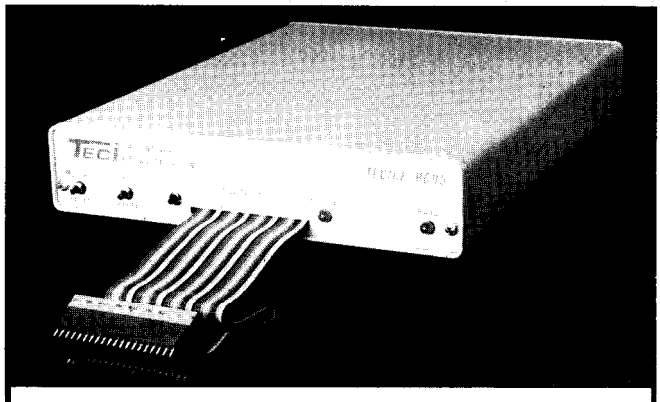


Figure 6—*The FSC evaluation software provides a window on all the FSC registers. which update automatically on the screen as information is analyzed.*

Tom Cantrell holds a B.A. in economics and an M.B.A. from UCLA. He owns and operates Microfuture Inc., and has been in Silicon Valley for ten *years involved in chip, board, and system design and marketing.*

IRS

222 Very Useful
223 Moderately Useful
224 Not Useful



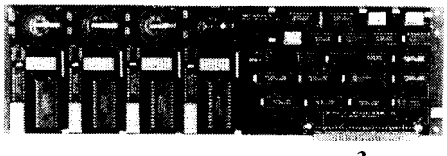
68HC05 AND 68HC11 IN-CIRCUIT EMULATORS

The TECICE-HC05 and TECICE-HC11 are low-cost real time in circuit emulators for the Motorola 68HC05 and 68HC11 families of single chip microcomputers. Any host computer with a serial port and terminal emulation software can be used with these emulators. Complete development system software is included for MS-DOS based computers. The base prices are $1195 for the HC05 and $995 for the HC11.

TECI THE ENGINEERS COLLABORATIVE, INC.
RR#3, BOX 8C . Barton, Vermont 05822
Phone (802) 5253458 . Fax (802) 525-3451

Reader Service #123

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



ROM
EEROM
SRAM
SRAM AS ROM

The Kolod Research R3OM Card

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

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

The R3OM card features include:

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

ngineering Excellence

Kolod Research Inc.
1898 Techny Court, Northbrook, Illinois 60062-5474 U.S.A.

$285.00 + shipping/handling
Made in U.S.A

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

Reader Service #135

CONNEC-TIME

Excerpts from the Circuit Cellar BBS

Conducted by
Ken Davidson

The Circuit Cellar BBS
300/1200/2400 bps
24 hours/7 days a week
(203) *871-1988*
Four Incoming Lines
Vernon, Connecticut

In ConnecTime for this issue, we'll be looking at what happens when a controller's power supply is glitched by the load it's controlling, the format of Intel hex files, what happens when bus drivers fight each other, and some alternatives to the discontinued TMS9918A. Let's begin, though, with a discussion about video digitizing.

Msg#:24677
From: FRANK HENRIQUEZ To: ALL USERS

Building a frame grabber is a major pain in the I/O port. So far, I'm using a Brooktree Bt253 RGB front end, an LM1881 sync separator, still-to-be-defined logic between the front end and the DRAM, and two NuBus chips to get the data off the board. Some questions:

I'm assuming that broadcast TV has a bandwidth of 3.58 MHz, so the maximum pixel resolution I could expect would be 320 pixels on a line or so. I also assume that most cameras have a higher bandwidth, maybe 4 or 5 MHz? Some commercial frame grabbers can digitize 640 pixels per line, or more. Are these digitizers just wasting effort digitizing a signal that, at best, really only has the equivalent of 512 pixels/line? Right now I'm debating between a 512 x 480 and a 640 x 480 resolution. The lower res is a lot easier to do. Any suggestions would be greatly appreciated!

Msg#:24698
From: JAMES D STEWART To: FRANK HENRIQUEZ

First, you must sample at two times the highest frequency of interest, per Nyquist theory. Assuming a monochrome signal, this would be a frequency of about 5 MHz and a sample rate of 10 MHz. If you are looking at color (as I assume you are since you mentioned RGB) it's a whole new ball game. The color info rides on a 3.579-MHz carrier and the usual procedure is to phase lock to that and sample at three times or four times that frequency.

Msg#:24712
From: FRANK HENRIQUEZ To: JAMES D STEWART

I learned some more about NTSC that may be of interest to anyoneeven thinking about frame grabbers. First, the color burst is not a problem; luminance info gets past it quite easily.. until it smacks into the sound trap. This limits the horizontal resolution of an NTSC signal to around 77-78 lines/MHz. Since the sound trap is at 4 MHz, this means the maximum resolution is going to be around 320 pixels-which explains why most home computers, like the C-64 and the Apple II, have a max resolution of 320.

I guess this means that if you plan on digitizing an RGB signal (three RS-170 signals, or the output of an NTSC-to-RGB converter) don't bother with anything greater than 320 pixels per line. This explains why the ImageWise looks fine, even though it only has 256 pixels per line-it's close to the NTSC limit. I'm now planning on building a 320 x 240 RGB frame (field) grabber; since it's in a 4:3 ratio, I'll get square pixels. Still, reality is disappointing at times-1 was hoping for 640 x 480.

I'm not sure if I need to sample at two times the signal frequency. Since the sound trap will pretty much kill everything above 4 MHz, and since the A/D front end has a limited bandwidth, I'm assuming that the combo will act as an antialiasing filter, but I may be wrong.

Msg#:24818
From: JAMES D STEWART To: FRANK HENRIQUEZ

Your points are well taken. I come from a professional video background, where video must move many times between units and a) never sees a sound trap until it is finally broadcast and b) must be as good as can be because of the degradation that creeps in. A point to consider is that in good sets, the sound trap is probably that: a trap or notch filter. You may still have picture detail getting through above the frequency of the trap.

Msg#:24837
From: DAVE EWEN To: FRANK HENRIQUEZ

...uh, sound is at 4.5 MHz, and I think all the video is below that.

Msg#:24844
From: MICHAEL JONES To: FRANK HENRIQUEZ

About two years ago I was working for a machine vision company. We were building a new system based on the VME bus. All our stuff was B&W, though we had three input channels so we could eventually upgrade to color. We had cameras that ran from

320 x 240 (noninterlaced) up through 768 x 576. They all put out RS-whatever standard signals. Our analog card could digitize pixels at up to -15 MHz, and our frame store would go up to 768 x512. I don't know about color cameras, but Iassumebynowthey are up around the same resolutions. At the time, though, cameras above about 512 pixels per line were pretty expensive. If you are outputting to a standard video monitor, then 640 x 480 is a real convenient resolution to aim for, since pixels end up being "square" on the monitor. Hope this was helpful.. .

Msg#:24847
From: FRANK HENRIQUEZ To: JAMES D STEWART

I picked 320x 240 just to get square pixels; I'll try to haveanoption to digitize 320 x 480, but that sounds too weird. TheTV Engineering Handbook says that NTSC is limited to 77-78 line pairs per MHz. If the sound trap is in the 5-MHz range, then the horizontal resolution is still under 400 pixels. And yet, as Michael Jones points out, there are frame grabbers digitizing at 640 (an higher); I assume that this is not standard NTSC, but RS-170 (is there frequency limit to RS-170 that would allow these higher resolutions?). I guess that I could go for 640 x 480 x 24 and just hope I can get hold of a camera or a video source that is capable of that resolution; otherwise, it's a waste of memory.. .

Msg#:24873
From: JAMES D STEWART To: FRANK HENRIQUEZ

As I understand it, what you refer to as NTSC is RS-170a. RS170 was the old monochrome spec, RS-170a defines the color encoding as well. There is only this standard and it exists first and foremost for the television broadcast industry. If you are interested, you can get the formal spec from EIA in Washington DC. If you want to build video equipment for the professional market, you must meet this spec or have a damn good reason not to. If, on the other hand, you just want something for experimenters that will work on a TV, you have a lot of freedom to play around. Another consideration is whether or not the signal will be videotaped. If so, It must be pretty **close** to RS-170a. I'll try to look up the video bandwidth issue tonight, but as I said earlier, I think the sound trap is a notch filter and that the video bandwidth can be higher than its notch frequency. I'll also give you a title of an excellent text on broadcast video engineering.

Msg#:24880
From: FRANK HENRIQUEZ To: JAMES D STEWART

Yeah! Books to wipe away ignorance (mine!). Apparently a lot of companies use RS-170 as a sync standard and little else, so 640 x 480 color (or mono) cameras are possible. So now I'm thinking about a 640 x 480 board again.

Msg#:24948
From: JAMES D STEWART To: FRANK HENRIQUEZ

Take a look at 'Television Broadcasting: Equipment, Systems, and Operating Fundamentals" by Harold E. Ennes, Howard W. Sams #20786. The Harold E. Ennes series of books are considered the Bible of the broadcast engineering business.

From video, let's go to something more heavy duty. When controlling large loads, it's very important to watch the controller's power supply when bugs crop up, as we see here.

Msg#:25006
From: DAYLYN MEADE To: ED NISLEY

Ed, here's the feedback I promised you. All the XBY addresses are OK. We have now narrowed down the problem and can reproduce the failure consistently. To review, the system crashes at a particular line with an "INVALID LINE NUMBER" error. All BASIC commands appear to be legal and the system logic sound, however the system seems to be more susceptible to crashing when jumping to line 4200 in the GOSUB 4000 call. The system is used to control an environmental chamber requiring switching of high-current resistive and inductive loads. The failure seems to occur while switching of the heating elements which the routine at 4200 controls. Here the chamber is within 10 degrees of the temperature setpoint and the elements are duty cycled to prevent overshoot. Our power supply seems to be well filtered, however we have not yet ruled out the possibility of power surges scrambling the micro. Reviewing the BASIC-52 manual, it seems Intel has had problems before with GOSUB and GOT0 vectors in Version 1 .O though we are using Version 1 .1. I have not been able to get any help from Intel and have called all over the country. Intel has informed us that they no longer support the BASIC-52 interpreter but instead make the source code available on a BBS. HELP! If anyone recognizes or has encountered this problem before, please respond.

Msg#:25183
From: ED NISLEY To: DAYLYN MEADE

If it isn't a memory problem caused by a glitch, I'll eat Serial Number 001 of that controller.. .

When you get the error, what happens when you try to list the program? If it doesn't get beyond the affected line, you've got corrupted memory. If it does, then you've got a transient.

An experiment is in order...rewire the system so you can control the high-current stuff with a switch on the line that ordinarily goes to the logic (that way you've got much the same connection as before and won't change the EM1 susceptibility a lot). Run a dummy program and flip the loads on and off a lot; if it blows up, you've found the problem. Come to think of it, you could write test program that does nothing but switch the loads on and off every few seconds, so if that fails you've got more information.

I really think you've got power problems rather than ROM problems, but the only way to find out is a few experiments!

Msg#:25279
From: DAYLYN MEADE To: ED NISLEY

Ed, you needn't eat your (or whomever's) serial number. It turns out the poor little 8052 was having some mental problems due to massive common-mode electroshock therapy. It seems that although the load being switched wasn't TOO big, the contactor (mercury wetted type) was emitting enough energy to cause 10V

p-p common mode on the 5-volt rail! The device IS shielded but the leads from the power supply to the shielded box aren't (or weren't). Thanks a lot for your suggestions (and to anyone else who pondered it as well) and comments. I workin 68k embedded systems (bare metal style) daily so if ever you have a need, drop me a line. I'm a newcomer to BBSing, so I don't check in all that frequently, but I am fairly regularly. Thanks again.

Msg#:25347
From: ED NISLEY To: DAYLYN MEADE

Whew! Hold the anchovies!

Note to all you readers out there: If you're doing industrial sorts of things, don't take _anything_ for granted! Once you've got the inevitable program bugs out, you can start working on the "once-in-a-while" hardware problems.

And _always_ verify that the power is clean and the logic signals logical!

Msg#:25653
From: DAYLYN MEADE To: ED NISLEY

Don't forget that the scope you are using may be seeing the nasties you are looking for on its ground plane too! In this case, it rendered the little nasty rather invisible and caused a great deal of grief. The last of the common mode gotchas, I expect. Thanks for the assistance.

A common format in which to put data for transfer to an EPROM programmer is Intel hex. In order to work with data formatted in such a way, you *have to know a little about the format itself.*

Msg#:24750
From: ED FANTA To: ALL USERS

I'm not sure how to read the Intel hex dump of the code I want to burn into an EPROM. If I am not mistaken the first byte on the line is the number of bytes on the line. I believe the next three bytes are the address, and the rest are the code to be burned. If this is true I might have a problem: the first line shows two bytes on the line but I see three in the dump. Am I not reading this right or did I have a problem with the download?

Msg#:24795
From: KEN DAVIDSON To: ED FANTA

In Intel hex, each line starts with a colon. The rest of the line is made up of pairs of digits, with each pair representing an 8-bit hex number. The number after the colon is the number of *data* bytes on the line. The next two numbers are the 16-bit address for the data on the line. The next number is the line type. A "00" means the line contains data; a "01" means the line is the last in the file. There are more types, but they aren't often used. After the line type comes the data. The last number in the line is a two's complement checksum of all the numbers in the line including the number of data bytes, the address, and the line type.

If all you want from the file is the data, you basically discard the first nine characters and the last two on each line.

Msg#:24843
From: ED FANTA To: KEN DAVIDSON

Thanks much. I did not know of the 00 byte or the end byte. That would explain why the burned EPROM and the downloaded code (that I was looking at with a word processor> seemed to be different.

We've *all accidentally configured expansion boards with conflictingportaddresses, but does does it really cause any harm? The next thread discusses just that.*

Msg#:25411
From: TOM CARTER To: ALL USERS

Can an I/O port clash cause permanent damage? This occurred when I installed an EPROM programmer card. I mistakenly assigned I/O on the card to the same area as the floppy controller. The system would not boot and I had to run the CMOS setup again since all of that information was lost. I don't think anything was damaged but I would like to know if it is possible. I am planning to start experimenting with I/O on a prototype card. Thanks a lot for any help.

Msg#:25413
From: NATHAN ENGLE To: TOM CARTER

I suppose that it's possible that you would damage the data transceivers on either card, since if both are enabled at the same time they would fight it out to see who can drive their signal the hardest, Typically with LS245s I wouldn't be too concerned unless you were planning to do it for an extended period of time. (The longer you go, the more times you're likely to have that condition in which one buffer says "1" [~2.4V or better] and the other one says "0" [as near to ground as possible]. When the two conflict the 0 will try to sink all of the l's current, possibly loading the 1 to the point that the pin burns out).

I'd be kind of surprised to see other signals that battle this way besides the **data** bus; maybe Ed will jump in to point out what I'm overlooking, but 1 think the data signals are the only ones that cards try to drive back onto the bus (at least for 8-bit cards).

Msg#:25467
From: ED NISLEY To: TOM CARTER

That's about the story. Somebody had a problem a while back with shared interrupts; turned out that one card had totem-pole outputs rather than an open-collector, so I suppose you could contrive a situation where that would burn out one or the other.. . but you'd have to work at it.

Msg#:26264
From: BRUCE GRAHAM To: NATHAN ENGLE

I'd be real surprised if two LS outputs fighting each other could burn a device out. A TTL totem pole output has a pull-up resistor to Vcc. The value varies according to family but is 50 ohms for an LS245. There are also one or more Vce drops which increase with current. I'd be surprised if you could get 24 mA flowing in this configuration. A more dangerous situation would be a 74AC245 and a 74LS245 shorted together, because the CMOS AC output will actually source a full 24 mA from the 5V rail. And it will probably lose the heating war that follows if the short is maintained .

Msg#:26298
From: NATHAN ENGLE To: BRUCE GRAHAM

I gotta agree with that; I just didn't want to rule out the possibility without knowing what's driving the bus in question. If the other card uses any sort of high-drive gate (like AC, or more likely 74F or 74AS in a PC) then I'd have to say that it's at least possible to burn out a gate.

Finally, the TI 9918A was a popular and useful display generator found in a number of past Circuit Cellar projects. The search for a replacement brought one user to the Circuit Cellar BBS.

Msg#:25202
From: MICHAEL FAIRMAN To: ALL USERS

A short while ago I asked a friend of mine who knew someone at Texas Instruments if he could get me some information about the 9918A Video Display Generator (there was a nifty Circuit Cellar article using it some number of years ago). To my dismay, he said that they had canceled the chip! Is this true, and if so, does anyone know of a similar device?

What I am looking for is cheap composite NTSC video (please, not RF modulated), preferably with direct access to the video RAM (something the 9918 did not allow). I am willing to have low resolution, such as the 256 x 192 pixels the 9918 put out-and would in fact prefer as few pixels as possible down to 128 x 128. I am also willing to use a reasonable amount of RAM to accommodate large pixel depths. If this is not already asking too much, I would prefer something with a color-lookup table.

If anyone can recommend a device which fits my needs, I would be quite grateful.

Msg#:25222
From: ROY CLAY To: MICHAEL FAIRMAN

MOS Technologies makes a chip that should meet your requirements. The 8563 Video Display controller can access up to 64K of video RAM. The screen resolution is programmable. It has NTSC and RGBI outputs. It has thirty-seven registers that control nearly every aspect of the video display. It should be available from Jameco or Kasara.

Msg#:25234
From: TIMOTHY TAYLOR To: MICHAEL FAIRMAN

Rockwell makes the 6549 color video display generator. I just started playing with it, so the jury's still out. It has RGB analog outputs (need a 1377 or 1378 to encode to NTSC), 256 x 210 x 4 resolution, 16 colors out of 4096, supports 4416 DRAMs for display storage, which can be accessed three ways:DMA, memory mapped, or I/O specifying x-y coordinates. Your local Rockwell app engineer will send you specs if you're interested. This is the same chip that's used in some NABTS teletext decoders.

Msg#:25257
From: MICHAEL FAIRMAN To: TIMOTHY TAYLOR

I appreciate the info about the 6549; although it has fixed resolution and a fairly small pixel size (I like to address bytes, assuming the x-y addressing is slower), it may be useful to look at.

The Circuit Cellar BBS runs on a 10-MHz Micromint OEM-286 IBM PC/AT-compatible computer using the multiline version of The Bread Board System (TBBS 2.1M) and currently has four modems connected. We invite you to call and exchange ideas with other Circuit Cellar readers. It is available 24 hours a day and can be reached at (203) 871-1988. Set your modem for 8 data bits, 1 stop bit, and either 300, 1200, or 2400 bps.

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 messaae areas is available on disk in two-month installments. Each installment comes&three 360K, 5.25" IBM PC-format disks and costs just $15. The installment for this issue of INK (June/July 1990) includes all public messages posted during March and April, 1990.

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

Circuit Cellar INK — Software (or BBS) on Disk
P.O. Box 772, Vernon, CT 06066

or use your MasterCard or Visa and call (203) 8752199. Be sure to specify the issue number of each disk you order.

STEVE'S OWN INK

Steve Ciarcia

Contemplation

It's spring now, and the softer feel of the wind lulls me into spells of daydreaming or, if I'm trying to justify spending the time, deep contemplation. We had a good spring rain last night, and the water dripping off the trees reminds me that another thunderstorm season is looming on the horizon. Over the last few years my automated abode and I have done our part to make America's insurance industry unprofitable. I hope that the lightning-strike problem has been solved, but maybe it's time to do some deep contemplating on whether or not this whole automated home project has been worthwhile.

Now it's true that my house is not your normal suburban colonial. The original sprawling house has grown to include a solarium, a green house, a couple of garages, and a lot of very fancy electronic and organic landscaping. I don't just sleep here, either. In addition to sleeping, eating, and all of the other activities normal folks do at home, I have the Circuit Cellar. I guess I'm trying to make the point that there are a lot of rooms on several different levels, a lot of different activities and, on occasion, a lot of people floating around the property. I turned to automation just to try to keep everything under control.

On the plus side, the automation does help my life go more smoothly. For a simple example, I give you lights. Since my electric bill is already higher than most businesses', I don't want to waste power by having lights on in rooms where there aren't any people. On the other hand, I often have my hands full when I walk through the house, and neither fumbling for a light switch with my elbow nor careening down a flight of stairs strikes me as a profitable way to spend time. Motion sensors and timing circuits make for a system that turns on the light when you enter a space, and turns the light off a few minutes after you leave. The system does what I want, and I don't have to think about lights any more.

I like my security and surveillance systems, too. I don't like the idea of someone rummaging around in my stuff when I'm not there, and a good automated security system (coupled with a host of very visible signs letting people know about the good automated security system) has worked to keep the peace. I've also become very fond of ROVER, my Remotely Operated Video Electronic Reconnaissance system. Anyone who has done extensive renovation knows that you spend a lot of time waiting for materials and workers to show up. With ROVER, I can come to the office and get some work done, then run back to the house when a truck shows up in the driveway. I don't know that it's perfect, but it beats the heck out of cooling my heels waiting for a load of sheetrock to arrive.

On the down side, my home automation system represents a whale of an investment. If you can forget the money that's gone into the system (my accountant won't let me), there are still hundreds of hours of my time wrapped up in the miles of cable and scores of circuit boards in the walls of my house. What's more, the investment doesn't end when the system is installed. There are always upgrades to be performed, bugs to be stomped, modifications to be planned, and mistakes to be corrected. When the occasional natural disaster comes along, it can mean days spent rewiring and rebuilding.

Another point on the down side is that casual visitors can "crash" my house. Most people can operate a standard light switch without getting into too much trouble. When they pick up a controller or programmer pod and start fooling around, though, the results can vary between expensive and terrifying. I still remember the time that an inquisitive four-year-old found an X-10 controller. Lights, stereo, and security all popping on and off.. .he was playing the controller like a piano. Now, when strange things happen after a party I tend to blame it on inquisitive (if incompetent) guests rather than itinerant ghosts. The first couple of times it happened, though.. .

All things considered, I'm glad I automated my house. I've learned quite a bit from the experience: Much that I've learned, I couldn't learn anywhere else. It has required dedication, though. It's required more dedication than most pets, and that's too much for most people. I guess it's just part of the price you get to pay for being a pioneer.

