EMBEDDED PC SPECIAL BONUS SECTION: EMBEDDED PC

CIRCUIT CELLAR INK®

THE COMPUTER APPLICATIONS JOURNAL

#75 OCTOBER 1996

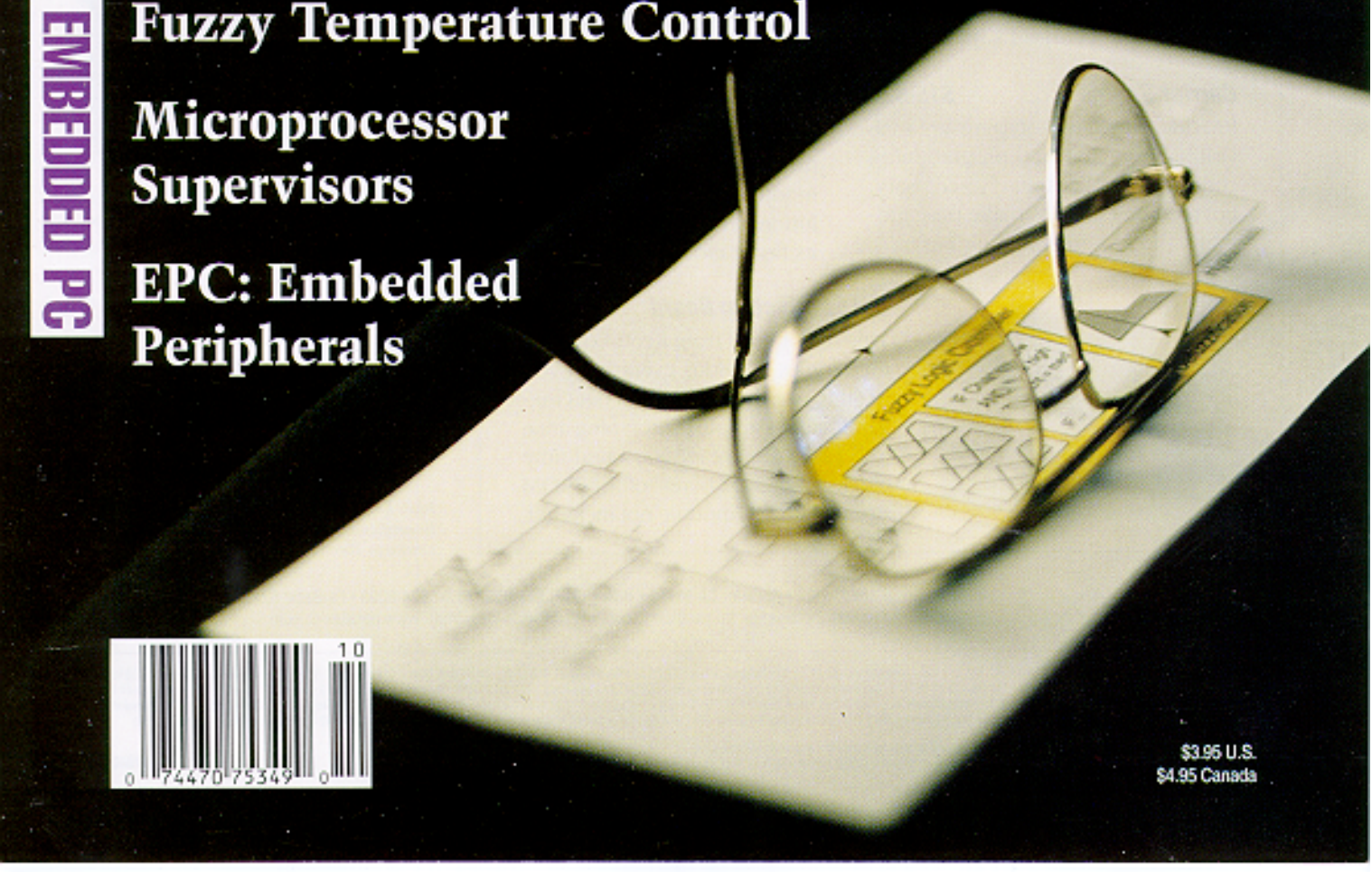
FUZZY LOGIC

Practical Fuzzy Logic Design

Fuzzy Temperature Control

Microprocessor Supervisors

EPC: Embedded Peripherals



0 74470 75349 0 10

$3.95 U.S.
$4.95 Canada

guest TASK MANAGER

Blowing Candles

I read recently that anniversaries—whatever their nature—have a significant impact on our lives emotionally and psychologically. People find themselves getting anxious as the anniversary of their good friend's fateful accident approaches. Or, people who spent much of their lives in academic circles find themselves getting down to business because it's September, a time they associate with a return to structure.

Well, it's October 1996 now-one year since I wrote my first guest editorial announcing the launch of Embedded PC, a "quarterly insert devoted to bringing you the latest on the embedded PC race."

And, what resonance does this hold for me?

Good question. I'm deeply satisfied that *EPC* has grown from two issues in '95 to six in '96 and eight in '97. I'm pleased that we've been able to identify significant developments in the industry and bring you editorial applying it. Similarly, I'm glad companies recognize that *EPC* fills an important niche and are taking advantage of its advertising opportunities.

But, a first-year anniversary also tells me the honeymoon is over. It's time to hunker down, solidify, make sure the foundation is solid. And, I'm here to tell you that we're already on it. 1997 will bring a broader array of embedded-PC companies and technology. Nouveau PC will keep you current with the latest products. Soon, *Embedded PC* will have its own corner of *INK*'s Web site. It'll be a lot of work, but we're strapping on fatigues.

Not surprisingly, this anniversary issue marks some significant milestones. On the real-time front, Naren Nachiappan tells us what's happening with Windows NT, and Mike Baylis shows us how to embed QNX in flash. Rick tells us about PCI coming to PC/104, and Fred conjures in the world of embedded peripherals.

INK Features zoom in on fuzzy logic. After Chris Sakkas's introductory article, Constantin von Altrock shows us two fuzzy-logic control applications-one for a thermostat and the other for a crane. David Prutchi finishes up on biomedical instrument interfacing, while David Rector offers Part 2 in a series on EPLDs.

For our columns, Ed powers up Zerobeat, Jeff runs a pedometer, and Tom sets the watchdog.

Unfortunately, this anniversary also marks a closure. Ed, who has been with us through 75 issues of Circuit Cellar INK, is moving on to other things. His projects have inspired many of you, and he will be sorely missed. He promises, however, to develop some of his favorite columns into books. So no doubt, we haven't seen the last of Ed.

In his place, *INK* will be opening MicroSeries, a column of multipart articles on specific topics. The column launches in December with a two-parter on home automation. Let us know about any software or hardware issues you'd like to delve into more deeply.

Janice

janice.marinelli@circellar.com

CIRCUIT CELLAR®

THE COMPUTER APPLICATIONS JOURNAL

FOUNDER/EDITORIAL DIRECTOR
Steve Ciarcia

EDITOR-IN-CHIEF
Ken Davidson

MANAGING EDITOR
Janice Marinelli

TECHNICAL EDITOR
Elizabeth Laurençot

ENGINEERING STAFF
Jeff Bachiochi & Ed Nisley

WEST COAST EDITOR
Tom Cantrell

CONTRIBUTING EDITORS
Rick Lehrbaum
Fred Eady

NEW PRODUCTS EDITOR
Harv Weiner

ART DIRECTOR
Lisa Ferry

PRODUCTION STAFF
John Gorsky
James Soussounis

CONTRIBUTORS:
Jon Elson
Tim McDonough
Frank Kuechmann
Pellervo Kaskinen

PUBLISHER
Daniel Rodrigues

PUBLISHER'S ASSISTANT
Sue Hodge

CIRCULATION MANAGER
Rose Mansella

CIRCULATION ASSISTANT
Barbara Maleski

CIRCULATION CONSULTANT
John S. Treworgy

BUSINESS MANAGER
Jeannette Walters

ADVERTISING COORDINATOR
Dan Gorsky

CIRCUIT CELLAR INK®, THE COMPUTER APPLICATIONS JOURNAL (ISSN 0896-8985) is published monthly by Circuit Cellar Incorporated, 4 Park Street, Suite 20, Vernon, CT 06066 (860) 875-2751. Periodical rates paid at Vernon, CT and additional offices. One-year (12 issues) subscription rate U.S.A. and possessions $21.95, Canada/Mexico $31.95, all other countries $49.95. All subscription orders payable in U.S. funds only, via international postal money order or check drawn on U.S. bank.

Direct subscription orders and subscription related questions to Circuit Cellar INK Subscriptions, P.O. Box 698, Holmes, PA 19043-9613 or call (800) 269-6301.

POSTMASTER: Please send address changes to Circuit Cellar INK, Circulation Dept., P.O. Box 698, Holmes, PA 19043.9613.

Cover photography by Barbara Swenson

PRINTED IN THE UNITED STATES

For information on authorized reprints of articles, contact Jeannette Walters (860) 875-2199.

HAJAR ASSOCIATES NATIONAL ADVERTISING REPRESENTATIVES

NORTHEAST & MID-ATLANTIC	MIDWEST & SOUTHEAST	WEST COAST
Barbara Best	Christa Collins	Barbara Jones & Shelley Rainey
(908) 741-7744	(954) 966-3939	(714) 540-3554
Fax: (908) 741-6823	Fax: (954) 965-6457	Fax: (714) 540.7103

Circuit Cellar BBS—24 Hrs. 300/1200/2400/9600/14.4k bps, 8 bits, noparity, 1 stop bit, (860) 871-1988; 2400/9600 bps Courier HST, (860) 871-0549. World Wide Web: http://www.circellar.com/

EMBEDDED PC

See Pages 63-90 for our Special Bonus Section

INSIDE ISSUE 75

READER I/O

OOPS!

The resitive construct shown in Figure 3 in "A/D Conversion with Zilog's Z8 Microcontroller" (INK 7 **1)** is not an R-2R DAC. Every individual bit has exactly the same effect on the output voltage.

I didn't look at the code to run the DAC. It may appear to work in that the output of the resistor array does change with the input code, but it doesn't change in the manner implied in the article (i.e., with the count progressing from 0 to 255). The output is not monotonic.

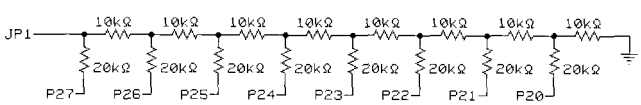
Tom **Bohley**
Colorado Springs, CO

INK *is lucky to have such attentive readers. Thanks for noticing.*

And, you're absolutely right. The resistor network doesn't work as the article or its author implied. We made the error, not Don.

Here's Don's revised resistor array.

Janice Marinelli



Contacting Circuit Cellar

We at *Circuit* Cellar *INK* encourage communication between our readers and our staff, so we have made every effort to make contacting us easy. We prefer electronic communications, but feel free to use any of the following:

Mail: Letters to the Editor may be sent to: Editor, Circuit Cellar INK, 4 Park St., Vernon, CT 06066.

Phone: Direct all subscription inquiries to (800) 269-6301. Contact our editorial offices at (860) 8752199.

Fax: All faxes may be sent to (860) 872-2204.

BBS: All of our editors and regular authors frequent the Circuit Cellar BBS and are available to answer questions. Call (860) 871-1988 with your modem (300–14.4k bps, 8N1).

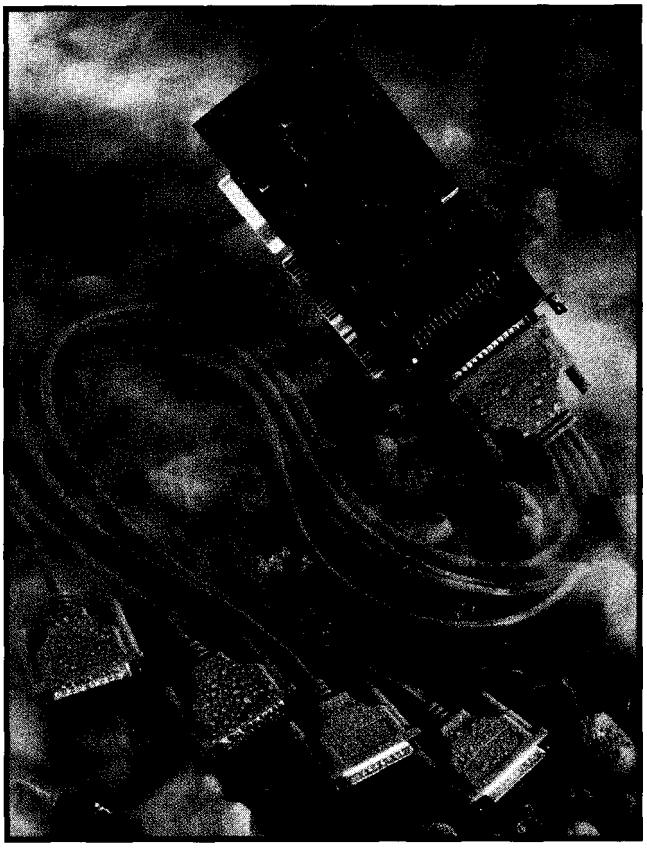
Internet: Letters to the editor may be sent to **editor@circellar.com**. Send new subscription orders, renewals, and address changes to **subscribe@circellar.com**. Be sure to include your complete mailing address and return E-mail address in all correspondence. Author E-mail addresses (when available) may be found at the end of each article. For more information, send E-mail to **info@circellar.com**.

WWW: Point your browser to **http://www.circellar.com/**.

FTP: Files are available at **ftp://ftp.circellar.com/pub/circellar/**.

NEW PRODUCT NEWS

Edited by Harv Weiner



QUAD 16550 SERIAL I/O CARD

Quadport is a 16-bit ISA/EISA product that features four high-speed 16550 serial ports compatible with DOS, Windows 95, Windows NT, and OS/2. A single 3' Quadplex cable harness (DC37M–DE9M or DB25M ends) interfaces with the serial devices. The product is completely free of jumpers for easy configuration of its I/O address and interrupt (IRQ) settings.

Quadport offers flexible I/O address mapping of its 16550 UARTs with 16-byte buffers to any location in the 64-KB I/O map. Interrupts may be routed individually or in IRQ share mode to 3, 4, 5, 6, 7, 9, 10, 11, 12, or 15. The high-speed serial ports with full modem-handshaking support are ideal for use with 9600+ bps external modems, pointing devices, or fast data transfer.

Quadport is a quick and efficient solution to adding up to four high-speed serial ports using the RS-232E interface to a computer system. Quadport sells for approximately $400 and comes with a three-year warranty.

Axxon Computer Corp.
3979 Tecumseh Rd. E • Windsor, ON • Canada N8W 1J5
(519) 974-0163 • Fax: (519) 974-0165
http://www.softio.com/ • axxon@mnsi.net

#500

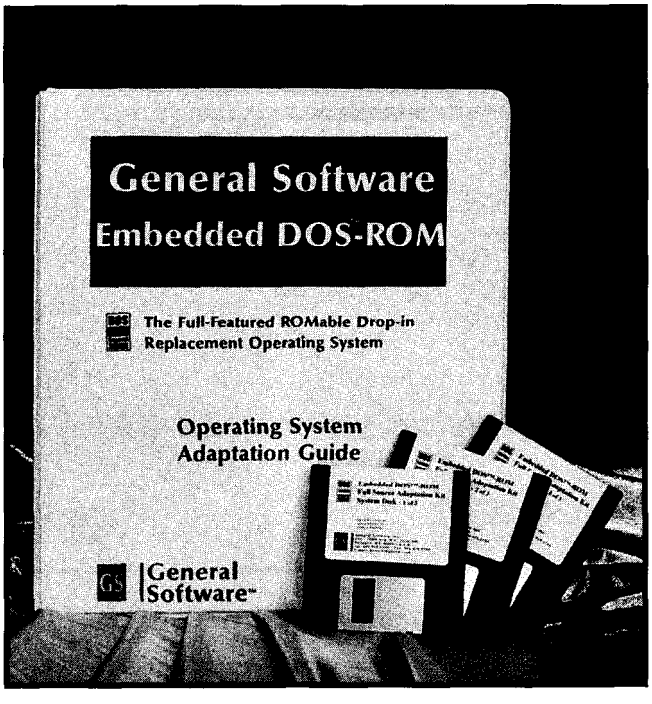
EMBEDDED DOS

The Embedded **DOS-**ROM occupies less than 32 KB of ROM and is a drop-in replacement operating system for embedded systems employing a more basic MS-DOS or ROM-DOS system software. A built-in ROM disk compatible with MS-DOS or ROM-DOS disk images finds them automatically without additional configuration. Applications include consumer'electronic commodity products such as battery-operated devices, cellular phones, Internet appliances, and smart pagers.

OEM developers can obtain a standard binary kit containing the Embedded DOS-ROM operating system and locator utility for $495. Source code is offered at an additional $995 for developers who wish to tailor the code to meet special requirements. Licenses are available for less than $6 per copy, depending on volume.

General Software, Inc.
320 108th Ave. NE, Ste. 400
Bellevue, WA 98004
(206) 454-5755
Fax: (206) 454-5744
http://www.gensw.com/general/

#501



NEW PRODUCT NEWS

INTELLIGENT UNIVERSAL PROGRAMMER

The new **Dataman-48** programmer features a 48-pin ZIF socket using Dataman's PinSmart technology. PinSmart enables the programmer to support all dual in-line devices up to 48 pins without additional adapters or converters. PinSmart also detects when devices are damaged or are making a bad contact on the programmer socket and halts operation before potential costly damage occurs. Using this highly flexible pin-driver system, Dataman-48 can do the job of several device-specific programmers at a far lower cost.

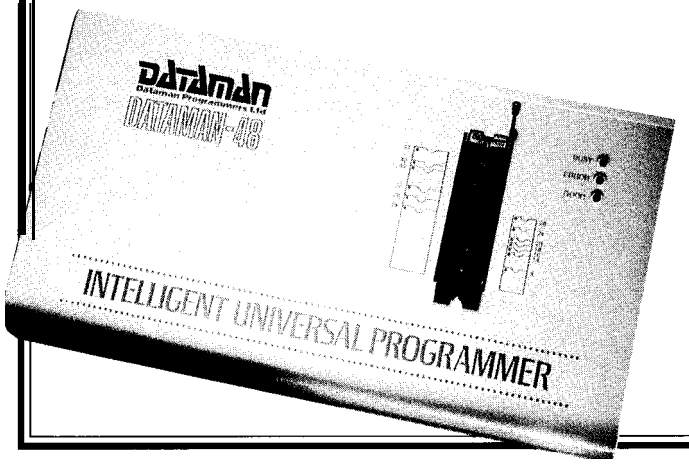
Each programmer comes standard with both DOS- and Windows-based software versions. Lifetime software updates are available on Dataman's BBS and Web page, and lifetime technical support is also free.

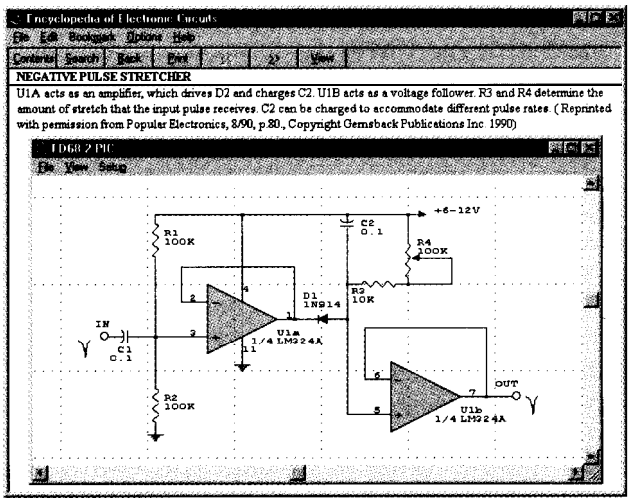
Dataman-48 was designed primarily as a development programmer. But, by using a special function with the PC software, you can turn it into a production programmer capable of making multiple copies of devices. In this so-called mass-production mode, copies of devices are made by simply inserting a device into the ZIF socket and closing the lever. Dataman-48's PinSmart technology detects when a good contact is made on all pins and starts programming automatically. Dataman-48 can also insert an incremental serial number into each programmed device.

At $1,249, Dataman-48 comes complete with power cord, parallel cable, DOS and Windows software, user manual, and a universal 44-pin PLCC adapter.

Dataman Programmers, Inc.
22 Lake Beauty Dr., Ste. 101
Orlando, FL 32806
(407) 649-3335
Fax: (407) 649-3310
http://www.dataman.com/

#502





ENCYCLOPEDIA OF ELECTRONIC CIRCUITS

The Encyclopedia of Electronic Circuits is a CD-ROM containing 1000 circuit designs taken from industry leaders such as Motorola, Texas Instruments, General Electric, RCA, National Semiconductor, and others. Developed by Mental Automation, the CD-ROM is based on McGraw-Hill's popular five-volume book series. The circuits were selected by the original author of the series, Rudolf Graf.

The CD-ROM enables a user to quickly find circuit designs by category or alphabetically. There are 142 circuit categories that include audio circuits, display circuits, measurement circuits, power supplies, radio circuits, and signal-generation circuits, among others. Once a topic is found, a text description of the circuit is provided, and the user can view and print out the complete circuit diagram using a built-in schematic viewer. Many schematics also include mechanical details of parts, IC packages, and waveform diagrams.

All schematics are redrawn in a vector format so they can be edited using Mental Automation's optional Super-CAD schematic editor. Output is available in AutoCAD DXF format. Many circuits can be transformed into PCBs using the printed-circuit-design software included with the CD-ROM. A list of 100 PCB projects is provided in both the printed and free online documentation.

The encyclopedia sells for $99 and runs under Windows 3.1+, including Windows 95. It runs on IBM compatibles with 4 MB of RAM, a Microsoft-compatible mouse, and graphics adapters supported by Windows. A CD-ROM drive (speed 2x or greater) is required.

Mental Automation, Inc.
5415 136th Pl. SE • Bellevue, WA 98006
(206) 641-2141 • Fax: (206) **649-0767**
http://www.mentala.com/ **#503**

NEW PRODUCT NEWS

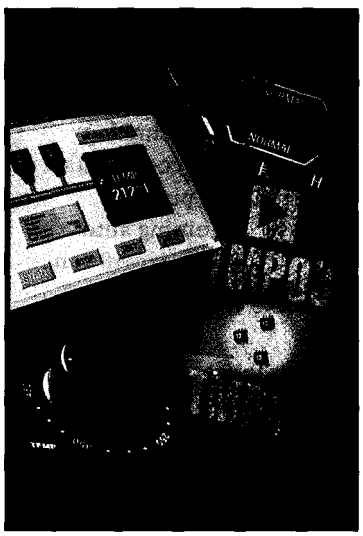
DIGITAL THERMOMETER IC

The **TMP03** and **TMP04** digital-output temperature sensors offer a complete and low-cost sensor-system on a chip in a 3-pin TO92 package. These highly integrated monolithic thermometers include all necessary functions (e.g., sensor, A/D converter, voltage reference, and control logic) to simplify complex thermal-monitoring systems. No extra user calibration, linearity compensation, or offset and drift adjustments are needed.

Accuracy over its -40" to + 100°C measurement range is guaranteed at less than ±3° when using a single +5-V supply. The ICs feature a PWM digital output where the high and low periods, T1 and T2, are proportional to the absolute temperature.

The time-domain digital output of the TMP03 and TMP04 is ideal where EMI, noise and distortion, long cable lengths, and other variables compromise analog signal linearity. The TMP03 features an open-collector 5-mA output for data going to PWM circuitry. The TMP04 provides TTL-CMOS-compatible outputs for direct interface with DSPs and microcontrollers.

The ratiometric structure of the IC's output enables decoding that is independent of clock drift and other sources affecting the accuracy of voltage-to-frequency converters and comparable devices. Another attribute of the TMP03 and TMP04 is their low 6.5-mW power consumption. While operating from a +5-V power source, their supply current (unloaded) is guaranteed less than 1.3 mA. Prices start at $2.49 in 1000-piece quantities.



Analog Devices, Inc.
804 Woburn St.
Wilmington, MA 01887
(617) 937-1428
Fax: (617) 821-4273

#504

FEATURES

FEATURE ARTICLE

Chris Sakkas

An Introduction to Fuzzy Logic

Chris sets the stage for this issue with a clear informative intro to fuzzy logic. In his opinion, fuzzy logic promises to add control to systems that have been difficult to model using a classical set theory.

Fuzzy logic is a new and growing technology that promises to add control to previously ill-behaved systems that have been difficult to model in the past.

As with many new technologies, fuzzy logic has struggled to gain acceptance. Nevertheless, it's an emerging paradigm that's being put to many new uses.

COMPARING THEORIES

In classical set theory, an element is either completely included or excluded from a set, known as a crisp set. This binary representation is the foundation of traditional logic. Only full or null membership in a set is recognized, and an assertion is either true or false.

Set operators-such as AND, OR, and NOT-formulate a binary output for multiple set elements. Boolean logic has been used in countless applications and control systems. It's the core of all digital computers and programming languages.

However, crisp logic cannot express imprecision. Fuzzy set theory recognizes that there are few areas where crisp sets actually exist.

Fuzzy logic offers partial set membership. Elements gradually transition from full to fully-not membership. A partial member is also partially not a

member. So, an element can be a partial member of more than one set.

Fuzzy logic makes up for the weaknesses of traditional logic. Systems not clearly defined, nonlinear, or imprecise can be modeled using fuzzy logic.

It offers fault tolerance and the ability to provide accurate responses to ambiguous data. Products designed with fuzzy logic have simpler controls, are easier to build and test, and provide smoother control than those using conventional systems.

This multivalued logic was first explored over 60 years ago by Polish logician Jan Lukasiewicz. Max Black, a quantum philosopher, continued work in this area and is credited with creating fuzzy-set membership functions.

However, it wasn't until Lotfi A. Zadeh, an electrical engineering professor at Berkeley, published "Fuzzy Sets" that fuzzy-set theory was fully developed.

Zadeh worked with complex, nonlinear systems. He discovered the more complex a system, the less meaningful low-level details are in describing overall system operation. He then attempted to describe a system's operation linguistically instead of mathematically.

Fuzzy logic is practical when one or more of the control variables are continuous. It is also useful when a mathematical model does not exist, is too complex to encode, or too complicated to be evaluated by a real-time system.

It's useful when high ambient noise levels must be dealt with or when you need inexpensive sensors or microcontrollers. When an expert is available who can define the rules for system behavior and the fuzzy sets that represent the characteristics of each variable, fuzzy logic is worthwhile.

USING FUZZY LOGIC

The fuzzy-logic procedure analyzes and defines a problem, creates sets and logical relationships, converts information to the fuzzy domain, and interprets the model to be used.

To use the fuzzy-logic model, define a problem with membership functions and convert it into a fuzzy-logic rule. Then, set up a procedure for fuzzifying and processing the problem and interpreting the rules. Finally, fine-tune the model for appropriate results.

A fuzzy set consists of elements that display a degree of membership in a set ranging from 0 and 1, inclusive. Figure 1 presents a fuzzy set of temperature values. Linguistic variables define sets such as Cold, Comfortable, Warm, and Hot. A temperature is an element of one or more of these sets with a value between 0 and 1.

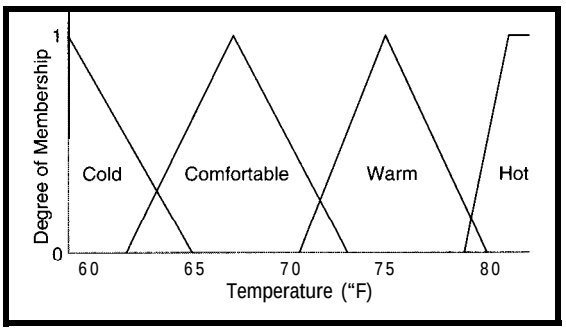


Figure 1--Four subsets of Temperature are defined in this graph. A single temperature element may have a degree of membership between 0 and 1 in more than one subset (e.g., 63°F is a member of both the Cold and Comfortable subsets. but has a greater degree of membership in the Cold subset).

Depending on you, 71°F has a 0.4 degree of membership in the Comfortable set and 0.2 in Warm. This method of using sets is more expressive than conventional logic, which defines 71°F as entirely in Comfortable, a range of 65–74°F.

FUZZY STAGES

A fuzzy system consists of fuzzification, inference, and defuzzification. During fuzzification, crisp numerical inputs are converted into fuzzy input values by using defined fuzzy sets.

Inference is made by using fuzzy rules and system input values to produce a numeric output for the rules. Defuzzification produces a single crisp value as the system output.

Fuzzy rules consist of if/then-type statements that are evaluated during inference. A fuzzy rule might be, "If Temperature is Warm, then make Fanspeed Medium."

A fuzzy system consists of many rules, some of which oppose others. One fuzzy rule often replaces many conventional rules. In inference, all fuzzy rules are evaluated in preparation for the next stage.

Since more than one rule may apply for any set of inputs, fuzzy logic must be able to combine the results of several rules. In defuzzification, multiple sets produce one single crisp output.

In the fan example, output may be an appropriate voltage to power the fan. You can combine results by performing a center-of-gravity computation on the maximum set-membership values of the active consequents. This technique is one of the most common.

FUZZY REAL WORLD

These three stages are used in all fuzzy-logic applications. An example of a real-life system is a ferry-boat controller. This system must decelerate a ferry to near-zero speed by the time it reaches the energy-absorbing barrier at the dock.

Three fuzzy sets are defined for velocity, distance, and RPM. Subsets handle the ranges required of such a system (e.g., the distance set-defined subsets for Near, Close, and Far).

Out of the seven fuzzy rules proposed for this system, many could decelerate or accelerate the boat. A single output variable is created by computing a center-of-gravity on the maximum set-membership values of the active consequents. The fuzzy-logic-based control system provides an appropriate solution for the problem and accommodates any velocity-versus-distance profile.

Another example of a fuzzy-based system is Japan's Sendai Metro. This automated railway, which opened in 1987, uses fuzzy-logic technology to increase accuracy in stopping at a platform, to provide greater rider comfort through smoother acceleration and braking, and to lower electric-power usage. This railway system sparked interest in fuzzy logic and made Japan a leader in fuzzy technology.

Since fuzzy logic is based in mathematics, it can be implemented in various ways. Depending on the application, fuzzy logic can use traditional microcontrollers or special-purpose fuzzy-logic systems. These systems provide faster inferencing since they are optimized for this function.

Also, many desktop computer applications have been generated that use fuzzy-logic technology or that allow developers to implement the technology in their own applications.

Recently, fuzzy logic has combined with other technologies. Combining fuzzy logic and neural networks improves speed and adaptiveness. A neural network converts knowledge into fuzzy rules and membership functions, and fuzzy logic optimizes the number of rules the neural network learns.

Since fuzzy logic excels in getting exact results from imprecise or ambiguous data, it's useful in industrial, commercial, and business applications.

In industry, fuzzy logic has made its biggest impact in controls. Systems too complex to model mathematically use fuzzy logic.

In nonlinear systems, fuzzy controllers perform functions previously requiring a human operator. The Omron Electronics temperature controller uses fuzzy logic and claims to be as much as 50% faster than conventional temperature controllers.

Fuzzy controllers also are easier to set up than conventional controllers. A lower cost micro can be used since fewer lines of logic are used, complex mathematical models do not have to be developed, and less computationally expensive programs are generated.

Fuzzy logic allows control systems to be developed faster and more cost effectively. And, it requires less maintenance over conventional controllers.

FUZZY WELCOME MAT?

Fuzzy logic will soon enter our homes, too. It's being used in the development of many consumer electronic devices and home appliances. Refrigerators, air conditioners, washing machines, and other appliances will benefit from fuzzy-logic control. These advances should improve appliance efficiency.

The Japanese have made great strides in their use of fuzzy logic in consumer electronics. They commonly use fuzzy-logic techniques in designing control circuits for TVs, VCRs, and camcorders.

Fuzzy-logic-based expert systems and simulations have also made an impact in business. Expert systems have been used for years in business to answer complex questions. Fuzzy logic-based expert systems are easier to implement and require fewer rules.

In fact, one example showed that the majority of probabilistic expert systems could be implemented as fuzzy-logic-based expert systems using only one-tenth the number of rules. Systems based on fuzzy logic allow computers to simulate human decision-making better than other methods that have been tried.

Fuzzy logic is a technology destined to find greater use in the coming years. Many scientists and engineers are beginning to see the fuzzy-logic advantages in many applications. ❑

Chris Sakkas is president of ITU Technologies, a company specializing in development tools for microcontrollers. You may reach him via E-mail at *chris@itutech.com* ***or telephone at (513) 574-7523.***

REFERENCES

J. Barron, "Putting fuzzy logic into focus," ***BYTE, 18,*** 111-I 18, 1993.

F. Bartos, "Fuzzy logic is clearly here to stay," ***Control Engineering, 39, 45-47, 1992.***

D. Brubaker, "Fuzzy-logic basics: Intuitive rules replace complex math," ***EDN, 37,*** 11 I-I ***16, 1992.***

D. Conner, "Designing a fuzzy-logic control system," ***EDN, 38, 77-88, 1993.***

E. Cox, "Fuzzy fundamentals," ***IEEE Spectrum, 29, 58-61, 1992.***

G. Legg, "Microcontrollers embrace fuzzy logic," ***EDN, 38, 100-109, 1993.***

K. Self, "Designing with fuzzy logic," ***IEEE Spectrum, 27, 42-44, 1990.***

L. Zadeh, "Fuzzy logic," ***Computer,*** ***21***, 83–93, ***1988.***

IRS

401 Very Useful
402 Moderately Useful
403 Not Useful

Practical Fuzzy-Logic Design

The Fuzzy-Logic Advantage

In this article, Constantin shows us a situation where fuzzy logic solves a problem that traditional logic cannot. His intention is not to replace conventional techniques, however, just extend them.

FEATURE ARTICLE

Constantin von Altrock

Over the past few years, systems designers have heard some rather controversial discussion about a new design technology called fuzzy logic. Some disciples of fuzzy logic have pitched it as the magic bullet for every type of engineering problem, and they've made wild and unsupported claims.

Such behavior raises skepticism among control-engineering specialists, causing some to completely condemn fuzzy logic as a marketing fad of no technical value.

The discussion in Asia and Europe about fuzzy logic has been much less religious and has kept closer to its real benefits. Fuzzy logic is considered one of the many tools necessary to build systems solutions in a fast, cost-effective, and transparent fashion [1]. It's not more or less than that.

Previous *INK* articles have generated explicit questions in how fuzzy logic achieves superior performance compared to other engineering design techniques. My goal is to show you exactly that.

To illustrate how fuzzy logic solves real-world problems and to compare it head-to-head with conventional design techniques, I'll use the case study of the antisway crane controller shown in Photo 1. I selected this real-world application for a number of reasons.

Unlike other technical real-world applications, the works of a crane controller can be understood quickly.

Antisway control of a crane is a simple problem, but conventional engineering techniques have a hard time arriving at a practical solution. In contrast, fuzzy logic rapidly delivers a good solution which has been successfully implemented on many crane controllers.

In this article, 1'11 assume you're familiar with the concept of fuzzy logic. If not, refer to this issue's "An Introduction to Fuzzy Logic" by Chris Sakkas, to the last fuzzy-logic issue (*INK* 56), or to Fuzzy ***Logic and*** *Neuro-**Fuzzy Applications Explained*** [2].

SINGLE-VARIABLE CONTROL

Before I start with the case study, let me clarify some control-engineering basics. Closed-loop control mostly involves keeping a single figure con-

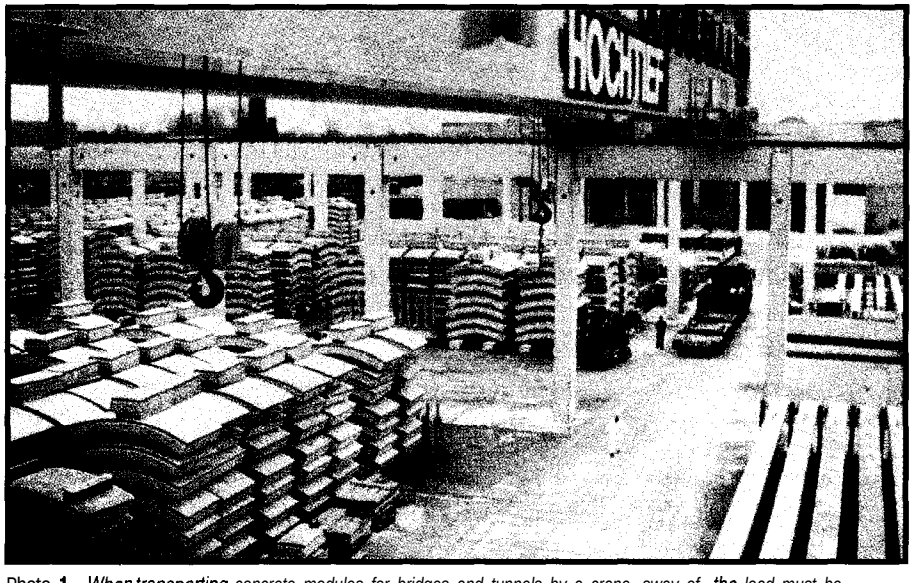


Photo 1—*When transporting concrete modules for bridges and tunnels by a crane, sway of the load must be controlled. A fuzzy-logic controller uses the experience of a human crane operator. This crane has two heads, each capable of transporting 64 tons, and it employs an antisway controller based on fuzzy logic.*

stant. It can be an electrical current, the mass flow of a liquid, or the temperature of a room.

To keep room temperature constant, a sensor delivers the value of the actual room temperature. The difference between this temperature and the desired room temperature is the temperature error.

The controller's objective is to get the temperature error to zero. To do this, it may turn a heater or air conditioner on or off. Such controllers are referred to as on/off-type controllers.

If the command variable of the process is continuous, such as with a gradual heat control, a so-called proportional-type (P) controller is used. This implements a control strategy that computes its output-the command variable of the process-by multiplying the error by a constant factor.

Many real-world processes involve a time lag. When controlling room temperature and the heater turns on, it takes a while before the heat reaches the sensor.

Because the controller gets the measured room-temperature variable with a time delay, it can overreact and apply more heat than what's required to get the temperature error to zero. This overreaction can cause unwanted oscillation of the room temperature.

To compensate for this effect, proportional-differential (PD) controllers add the time derivative of the error multiplied by a constant to the error.

Proportional-integral-differential (PID) controllers also consider the integral of the error and add it to the output signal. This method ensures that the error eventually reaches zero because even a small offset in the error value is compensated for as its integral raises to a high value over time.

Because conventional controllers only use the error or signals derived from the error as input and the command variable of the process as output, they are called single-loop controllers. These controllers are usually simple to program and easy to tune.

Replacing such controller types by a fuzzy-logic controller only delivers a better performance in cases where the conventional controller doesn't cope well with the nonlinearities of the process under control [2, p. 82].

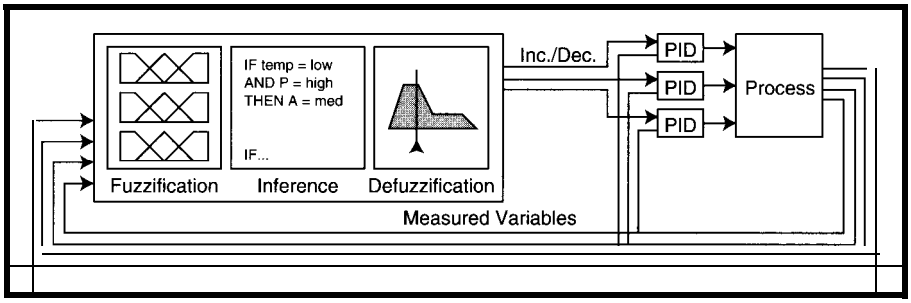


Figure 1—*In many applications, keeping a single figure of the process constant is relatively easy and can be done by conventional PID controllers. However, to optimize the operation point the supervisory controller modifies the set points of the PID controllers.*

MULTIVARIABLE CONTROL

A much higher potential for fuzzy logic lies in the fact that it also facilitates the design of multivariable control systems. In a multivariable control loop, the control strategy considers not only one input variable (e.g., error) but different types of measured variables.

In the case of the room thermostat, a multivariable control strategy considers temperature error and room air humidity to set the command variables of the heater and air conditioner.

To deal with multiple input variables, a controller must assume a mathematical model of the controlled process. For the multivariable room-thermostat control strategy, such a model can be the humidity and temperature relationship which describes comfortable conditions.

In most real-world applications, the mathematical relation between the different variables cannot be described so easily. So, multivariable control is not commonly used.

However, application areas exist where multivariable-control strategies must be used, such as continuous process control as found in the food and chemical industries. Keeping single figures of the processes constant is easy in most cases. Controlling the plant's optimal operation point involves multiple variables.

Because of the difficulties involved with deriving mathematical models, automated control is rare and human operators often adjust the set points of the individual PID controllers.

This type of multivariable control is also referred to as supervisory control because the multivariable control strategy analyzes and supervises the set points of underlying control loops.

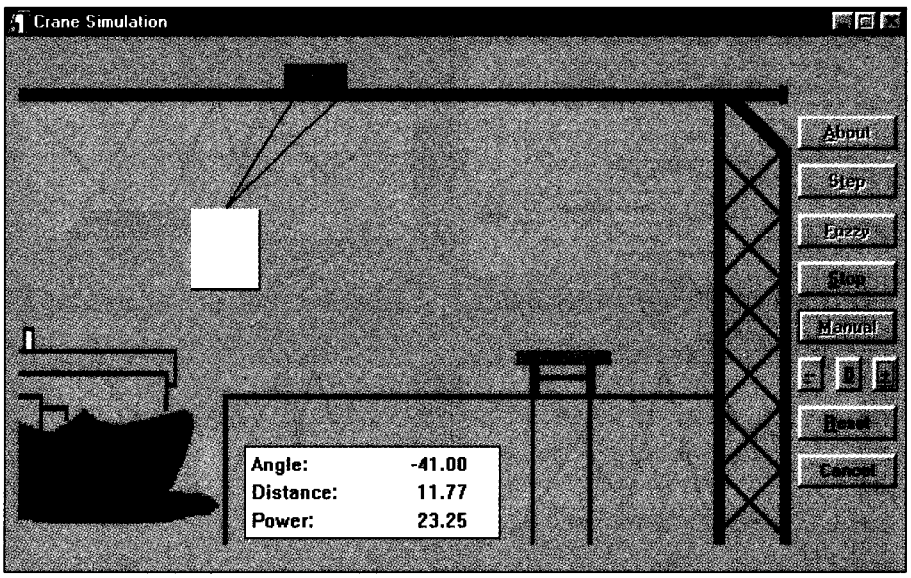


Photo 2—*The software simulation of container-crane operation visualizes the operation of the fuzzy-logic antisway controller. The fuzzy-logic control strategy is to get the container from the ship to the target as quickly as possible. When the target is reached, the sway must be compensated before releasing the container.*

Spreadsheet Rule Editor - RB1

	IF		THEN	
	Angle	Distance	DoS	Power
1	zero	far	1.00	pos_medium
2	neg_small	far	1.00	pos_big
3	neg_big	far	1.00	pos_medium
4	neg_small	medium	1.00	neg_medium
5	pos_small	close	1.00	pos_medium
6	zero	zero	1.00	zero

Photo 3—*Each row corresponds to a fuzzy rule. The columns Angle and Distance underneath IF form the conditions of the rules. Underneath THEN, the column Power forms the conclusion of the rules. The DoS (Degree of Support) column can contain an individual weight of each rule.*

In contrast to conventional design techniques, fuzzy logic enables the design of such multivariable control strategies directly from human-operator experience or experimental results (see Figure 1).

Using such existing knowledge and circumventing the effort of rigorous mathematical modeling, the fuzzy-logic design approach delivers efficient solutions faster,

The question therefore becomes, "How can operator experience be put into a fuzzy-logic system?" The following case study of a container-crane antisway control illustrates this.

CONTAINER-CRANE CONTROL

Container cranes load and unload containers to and from ships in most harbors (see Photo 2). They pick up single containers with flexible cables mounted at the crane head.

The crane head moves on a horizontal track. When a container is picked up and the crane head starts to move, the container begins to sway. While sway does not affect the transport, a swaying container can't be released.

There are two trivial solutions to this problem. One is to position the crane head exactly over the target position and wait until the sway

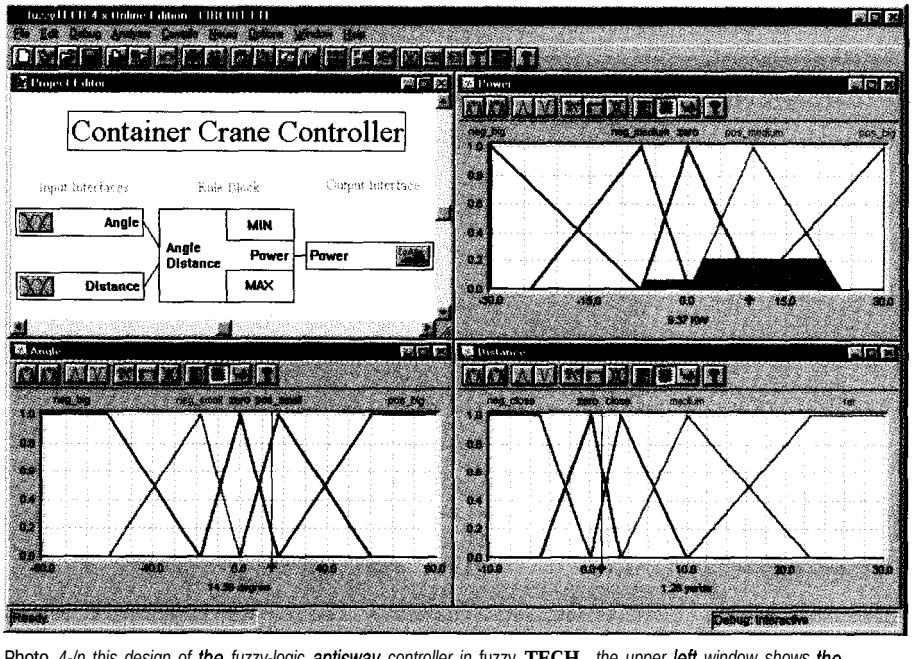


Photo 4—*In this design of the fuzzy-logic antisway controller in fuzzy* **TECH,** *the upper left window shows the structure of the system with two inputs, one output, and one rule block. The upper right window visualizes the defuzzification of Power, and the lower windows show the fuzzification of Angle and Distance.*

dampens to an acceptable level. On a nonwindy day, this eventually happens, but it takes far too much time. A container ship has to be loaded and unloaded in minimum time.

The alternative is to move the container so slowly that no sway occurs. Again, this technique works on a nonwindy day and takes too much time.

Another solution is to build container cranes where additional cables fix the position of the container during operation. This alternative is very expensive.

CONTROL-MODEL ALTERNATIVES

Many engineers have attempted to automate this control task via conventional PID, model-based, and fuzzy-logic control strategies.

Conventional PID control was unsuccessful because the control task is inherently nonlinear. For example, sway minimization is important only when the container is close to the target.

Others have tried to derive a mathematical model of the crane to use in a model-based controller. They came up with a fifth-degree differential equation that describes the mechanical behavior. In theory, a controller design based on this model works. In reality, it doesn't.

One reason for failure is that the weight of the container is unknown. Also, the crane-motor behavior isn't as linear as assumed in the model. Its gear box involves slack, its head only moves with friction, and its cables involve elasticity. As well, disturbances such as wind gusts aren't included in the model.

LINGUISTIC CONTROL STRATEGIES

On the other hand, a human operator can control a crane without differential equations. (Chances are, if the operator knew how to use differential equations, he wouldn't operate cranes.)

An operator doesn't even use the cable-length sensors that a model-based solution requires. Once the container is picked up, the operator starts the crane with medium motor power to see how the container sways.

Depending on the reaction, the motor power is adjusted to get the container a little behind the crane head. In this position, maximum speed is reached with minimum sway.

In approaching the target position, the operator reduces motor power or applies negative power. The container gets a little ahead of the crane head until the container almost reaches target position.

Motor power is then increased so the crane head is over target position and sway is zero. No differential equations are required, and disturbances and nonlinearities are compensated by the operator's observation of the container's position.

The operator's control strategy can be described by several rules:

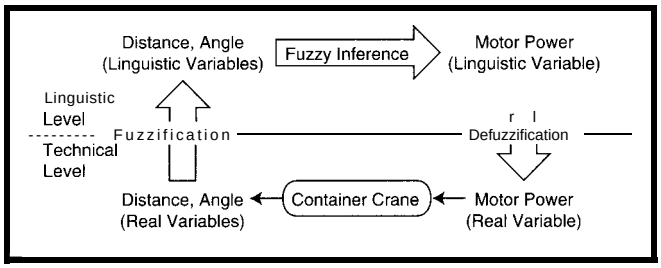


Figure 2-The complete *control loop* of *the fuzzy-logic* crane controller *has three* steps. The variables *Distance* and Angle measured at the crane are translated *into linguistic* variables *(fuzzification) and* used *to* evaluate *the condition* of the *fuzzy* rules *(fuzzy* inference). *A linguistic value for Power is translated back* into a *numerical value (defuzzification).*

- start with medium power
- if you're still far away from the target, adjust the motor power so the container gets a little behind the crane head
- if you're closer to the target, reduce speed so the container gets a little ahead of the crane head
- when the container is very close to target position, power up the motor
- when the container is over the target and the sway is zero. stop the motor

The advantage of fuzzy logic is that it can use the same sort of rules as the human operator.

To automate control of this crane, sensors are used for the head position (Distance) and the angle of the container sway (Angle). Using these inputs to describe the current condition of the crane, the rules can be translated into the if/then statements in the fuzzy-logic table in Photo 3.

The first condition describes the value of Distance, and the second the value of Angle. The conditions are combined by AND since both have to be valid for the respective situation.

Once you have rules describing the desired behavior of a system, the question becomes how to implement these rules. Consider using a programming language to code the if/then rules. Unfortunately, the conditions of the rules use words that need to be defined, and exact definitions don't exist.

However, fuzzy logic provides nonexact definitions for the words. Thus, you use linguistic rules for control-system design directly.

FUZZY-LOGIC CRANE CONTROLLER

Figure 2 shows the complete structure of a fuzzy-logic controller. All sensor signals have to be translated into linguistic variables. A measured distance of 12 yd. has to be translated to the linguistic value "still medium, just slightly far." This step is called *fuzzification* because it uses fuzzy sets for translating real variables into linguistic variables.

Once all input variable values are translated into linguistic variable values, the so-called fuzzy-inference step evaluates the if/then fuzzy rules that define system behavior. This step yields a linguistic value for the linguistic variable. So, the linguistic result for Power could be "a little less than medium."

Defuzzification translates this linguistic result into a real value that represents the power setting of the motor in kilowatts.

DESIGN AND IMPLEMENTATION

The development of fuzzy-logic systems involves specific design steps:

- design the inference structure—specify how the output variables connect to the input variables by the rule blocks.
- define the linguistic variables-the variables form the vocabulary used by the fuzzy-logic rules expressing the control strategy.

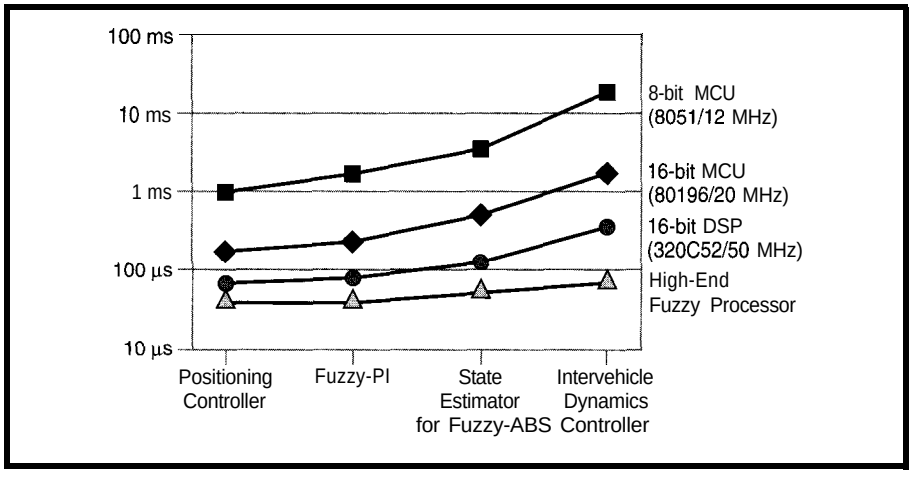


Figure 3—*Using* a set of standardized *benchmark systems, the performance of* different hardware platforms ***can*** *be* compared. *On a 80.51 MCU, a small fuzzy-logic system can be* computed in about a millisecond. *Faster microcontrollers and DSP can compute even very large fuzzy-logic systems in fractions of a millisecond.*

- create an initial fuzzy-logic rule base using all available knowledge on how the system should perform.
- debug, test, and verify the system offline for completeness and non-ambiguity-use a software simulation or sample data of the process if it exists in this step.
- debug online-connect the fuzzy-logic system to the process under control and analyze its performance in operation. Because fuzzy logic lets you modify the system in a straightforward way from the performance you observe, this step can expedite system design rapidly.

Most systems today are developed using software tools like the one in Photo 4. Such development tools not only support all listed design steps, but they also can generate the entire system for different hardware platforms. For microcontrollers (MCUs), the tools generate assembly code. For PLCs, they generate function blocks. For PC/workstations, they generate C code.

Many years ago, the computation of the fuzzy-logic algorithm was so inefficient on a standard MCU that dedicated fuzzy-logic processors were developed. Today, code efficiency has improved dramatically.

Figure 3 shows different MCUs, a DSP, and today's fastest fuzzy-logic processor computational performance for four benchmark systems. The time shown on the vertical logarithmic scale is the total time required to compute the complete fuzzy-logic system.

On a standard 12-MHz 8051 MCU, small fuzzy-logic systems compute in just one millisecond, and 16-bit MCUs can compute large fuzzy-logic systems in the same amount of time. This speed enables the integration of a fuzzy-logic system with most embedded system designs. However, some embedded system applications require even faster computation [e.g., antilock brakes in cars, ignition control, or hard-drive positioning).

To expedite the fuzzy-logic algorithm, some semiconductor manufacturers include specific fuzzy-logic instructions with their new-generation MCUs. For example, Motorola's 68HC12 MCU features a complete fuzzy-logic function set in assembly language.

LESSONS LEARNED

Fuzzy logic enables the use of experience and experimental results to deliver more efficient solutions. It does not replace or compete with conventional control techniques.

Rather, fuzzy logic extends the way automated control techniques are used in practical applications by adding supervisory control capabilities. The container-crane example demonstrates that fuzzy logic delivers a transparent, simple solution for a problem that's much harder to solve using conventional engineering techniques. ❑

Constantin ***von Altrock began research*** on fuzzy ***logic with*** *Hewlett-* ***Packard in*** *1984.* ***In 1989, he founded*** and ***still*** manages ***the Fuzzy*** Technologies ***Division of Inform Software, a market leader in fuzzy-logic development tools and turn-key applications. You may reach Constantin at*** *cva@* inform-ac.com

SOFTWARE

You may download the complete animated software simulation of the container crane for Windows from Circuit Cellar's BBS together with a simulation-only version of *fuzzy*TECH and the FTL source code of the crane controller. You may download the source code of the benchmark suite from *fuzzy*TECH's Web site (http://www.inform-ac.com).

REFERENCES

[1] C. von Altrock, "Fuzzy Logic Applications in Europe," in J. Yen, R. Langari, and L.A. Zadeh (eds.) ***Industrial Applications of Fuzzy Logic and Intelligent Systems***, IEEE Press, Chicago, 275–310, 1995.

[2] C. von Altrock, ***Fuzzy Logic and NeuroFuzzy Applications Explained,*** Prentice Hall, Englewood Cliffs, NJ, 1995.

SOURCES

68HC12 MCU
MCU Information Line
Motorola
P.O. Box 13026
Austin, TX 7871 l-3026
(512) 328-2268
Fax: (512) 891-4465
http://freeware.aus.sps.mot.com/

***fuzzy*TECH**
Inform Software
2001 Midwest Rd.
Oak Brook, IL 60521
(630) 268-7550
Fax: (630) 268-7554
http://www.inform-ac.com/

I R S

404 Very Useful
405 Moderately Useful
406 Not Useful

FEATURE ARTICLE

Constantin von Altrock

A Fuzzy-Logic Thermostat

It's easy to think that the old way is good enough, especially for everyday controls. But, Constantin shows us otherwise. A fuzzy-logic AC controller uses several variables to bring greater comfort at less cost.

The constantly increasing performance/price ratio of microcontrollers means electronic systems can replace more and more electromechanical ones. In design, the goal is not to just replace the solution, but also to improve it by adding new functionality.

One product where this goal has been achieved is the fuzzy-logic thermostat developed by Microchip Technology and Inform Software for an air-conditioning (AC) systems manufacturer. The design uses fuzzy logic to implement a radically new and efficient concept of a thermostat on a low-cost microcontroller.

The heating and cooling of homes and commercial buildings consume a significant portion of the total energy produced in the world. Increased efficiency in these systems can yield big energy savings.

These savings can be found in construction improvements (e.g., better insulation) or more intelligent building control strategies. Here, I'll focus on the latter-applying fuzzy-logic control techniques for AC systems.

Fuzzy logic offers a technical control strategy that uses elements of everyday language. In this application, it was used to design a control strategy that adapts to the individual user's needs. It achieves a higher comfort level and reduces energy consumption.

With a fuzzy-logic software development system, the entire system, which includes conventional code for signal preprocessing and the fuzzy-logic system, can be implemented on an industry-standard 8-bit microcontroller. Using fuzzy logic on such a low-cost platform makes this a possible solution with most AC systems.

AC CONTROL

Quite a few AC systems already use fuzzy-logic control. In 1990, Mitsubishi introduced their first line of fuzzy-logic-controlled home ACs. Industrial AC systems in Japan have been using fuzzy logic ever since. Now, most Korean, Taiwanese, and European AC systems use fuzzy logic.

There are different incentives to use fuzzy logic. For industrial AC systems, it minimizes energy consumption. The controller optimizes the set values for the heater, cooler, and humidifier depending on the current load state.

Car AC systems use fuzzy logic to estimate the temperatures at the driver's head from multiple indirect sensors.

Home ACs are much simpler. They don't contain a humidifier and only cool or heat at one time. Fuzzy logic gives them robust temperature control.

Each home AC has a thermostat that measures room temperature and compares it with the temperature set on the dial. The thermostat uses a bimetallic switch and compares the set temperature with room temperature. It minimizes the number of AC starts by using a hysteresis.

Figure 1 shows the control diagram of a straightforward microcontroller implementation of this principle. The difference between the set and actual room temperatures turns the AC on and off using a hysteresis stage.

Since most home AC systems don't provide continuous power control and the number of on/off switches is minimized, there's no real room for improvement in this design.

FUNCTIONAL ANALYSIS

The question is: how do you improve the design and add functionality.

A thermostat's sole purpose is to keep the room temperature constant. If the AC can only be turned on or off, not many design alternatives are obvious.

But, isn't this a major error in thinking? The original objective for a thermostat is not to keep temperature constant but to maintain maximum comfort.

Room temperature doesn't always correspond to the subjective temperature felt by people. During the day, a higher temperature is considered more comfortable than at night. The same room temperature is perceived as warmer if the room is sunlit.

Empirical analysis of how people adjust the temperature dial on their ACs tells even more. If the set temperature is turned down significantly at once, it's pretty obvious that a large cooling effect is desired.

To expedite the cooling, most people turn the dial down lower than what typically corresponds to a comfortable room temperature. Usually, they forget to put the temperature dial up again when a comfortable room temperature is reached. Before this error is corrected, the increased cooling wastes energy.

Another example is when the set temperature is changed only by a small amount. This indicates that the user wants the room temperature to be kept exactly at a desired point.

If the air conditioner overreacts, the user corrects the temperature again, seeking the desired room temperature. This both wastes energy and annoys the user.

INTELLIGENT THERMOSTAT

Obviously, a thermostat that understands and interprets these conditions and interdependencies can do a much better job than the simple on/off-type controllers shown in Figure 1.

However, putting such intelligence into a microcontroller's assembly program is anything but easy. The algorithm is based on empirical knowledge, and it involves many different factors and conditions. The design of a mathematical model is thus intractable.

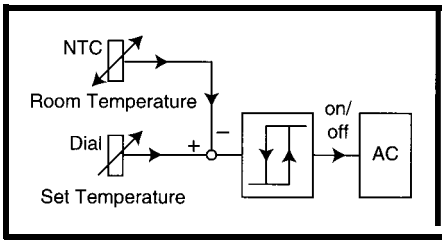


Figure 1-A conventional thermostat compares *room temperature with set* temperature *to turn the* AC on and Off.

Figure 2 shows the structure of the intelligent fuzzy-logic thermostat. The underlying design is the same as with Figure 1, but the fuzzy-logic controller intervenes at two points.

One output of the fuzzy-logic system corrects the set temperature, and another output adapts the hysteresis interval.

The input variables of the fuzzy-logic controller stem mostly from the set-temperature dial and the room-temperature sensor. An inexpensive LDR photo sensor measures the brightness in the room.

FUZZY -CONTROLLER DESIGN

Design, debugging, test, and implementation of the fuzzy-logic controller was supported by the development software *fuzzy*TECH[3]. The fuzzy-logic controller uses five input variables (computed from the sensory values) to analyze and interpret the conditions in the room:

- Number of set-temperature changes (ChangeNr)

This input signal identifies a user who tries to set the room temperature very precisely (rule 4 in the lower window in Photo 1). To satisfy such a user, the hysteresis is set to s ma 1 1.

This variable counts each time the user moves the dial. Every 6 h, this variable is counted down until zero is reached.

. Difference between the set and room temperatures (TempError)

When the difference between the set and room temperatures is very large, the fuzzy-logic system increases the temperature error (rules 5 and 6).

At the same time, the hysteresis is set to 1 a rge, so disturbances do not interrupt the cooling process. This strategy ensures the desired temperature is reached as quickly as possible.

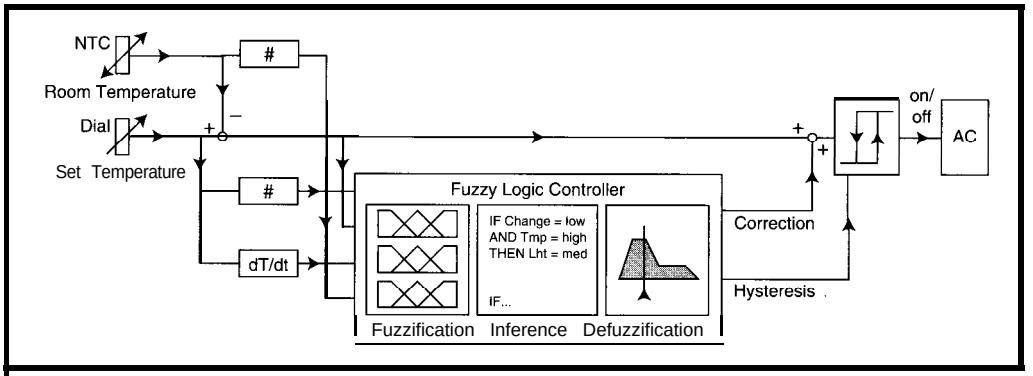
Attempting to alter temperature quickly is compromised by the fact that overshoot and undershoot are likely. However, with strong temperature errors, overshoot can be beneficial.

For example, if you come into a previously unused room where the AC was turned off and set the dial to a much lower temperature, the short undershoot of the room temperature gets rid of excessive heat stored in the walls and furniture of the room.

- Last set-temperature change (dTemp_by_dt)

The amplitude of the last set-temperature change indicates whether you want to have a strong cooling effect or to fine-tune the room temperature.

For example, rule 3 of the fuzzy-logic controller uses this input variable to detect fine-tuning. Because this signal is a differentiated signal, it disappears after 30 min. if the dial is unmodified.



igure 2—*The fuzzy-logic controller corrects both set temperature and hysteresis depending on the room usage and condition, hich is why a fuzzy-logic approach was selected for the implementation of the empirical know/edge on the microcontroller.*

- Number of room temperature changes > 3°F within past 2 h (RoomFluct)

This input variable indicates how heavily the room is used. Room temperature changes larger than 3°F are typically caused by open windows or by conferences involving an overhead projector and a large audience.

- Brightness in the room (Brightness)

If direct sunlight hits the room, the set temperature automatically reduces (rule 2) to increase comfort. During the day or when room lights are on, the set temperature increases slightly (rule 1) and the hysteresis is set to sma 11 to conserve energy.

CONTROL STRATEGY

Photo 1 shows the main window of the development software *fuzzy*TECH during design. The Project Editor window displays the structure of fuzzy-logic inference.

The fuzzy-logic thermostat's structure is straightforward. All input variables are fuzzified and fed into one rule block. The two outputs of the rule block become the outputs of the fuzzy-logic system and are then defuzzified.

The Correction window exemplifies a membership-function set definition for the linguistic output-variable definition. The grayed areas visualize the process of defuzzification.

Some of the fuzzy-logic controller rules are listed in the Spreadsheet Rule Editor window. Each row represents a fuzzy-logic rule, expressing part of the empirical knowledge implemented in the system.

The five left columns under the If button are input variables. Each field shows the value of the linguistic variable. The two double columns on the right represent the two output variables. The numbers between 0 and 1 in the smaller columns denote the relative weight of the rules that can be tuned during controller optimization.

All input variables have three terms with piecewise linear membership functions. The output variable Correction has five terms and uses center-of-area defuzzification. The output variable HysteresisS has three terms and uses center-of-maximum defuzzification.

In total, 34 rules describe the complete fuzzy-logic control strategy.

CHOOSING MICROS

*fuzzy*TECH generates the fuzzy-logic controller as complete program code from the graphical representation. The output is portable C source code, specialized-function blocks for PLCs, or assembly code for microcontrollers.

Because the fuzzy-logic thermostat is a low-cost, mass-market product, the PIC16C71, a standard 8-bit microcontroller, was used for the thermostat's prototype.

Since this microcontroller only provides 1024 words of program ROM and 36 bytes of RAM, it greatly limits the complexity of conventional assembly programs that can be implemented.

Using *fuzzy*TECH's PIC assembly-language generation, the fuzzy-logic

controller requires only 550 words of ROM and needs only temporary RAM storage. This arrangement enables you to implement your program on a low-cost microcontroller along with other periphery control code.

SIMULATION RESULTS

The fuzzy-logic system was tested using data recorded in rooms of different buildings under various conditions. The test data was preprocessed using an Excel spreadsheet.

To test the performance of the fuzzy-logic solution, *fuzzy*TECH's Excel Assistant was used. Spreadsheet cells link directly to the inputs and outputs of a fuzzy-logic system. Since this link is dynamic, the fuzzy-logic system can be monitored and modified using *fuzzy*TECH's analyzers and editors while browsing through the data sets.

Analysis of the controller performance shows that the fuzzy-logic thermostat detects situations where less cooling sufficed. In a standard residential house, average energy consumption was reduced by 3.5 %. As well, comfort level increased, since depending on the situation, the fuzzy-logic thermostat reduced the room temperature by up to 5°F more than the conventional thermostat.

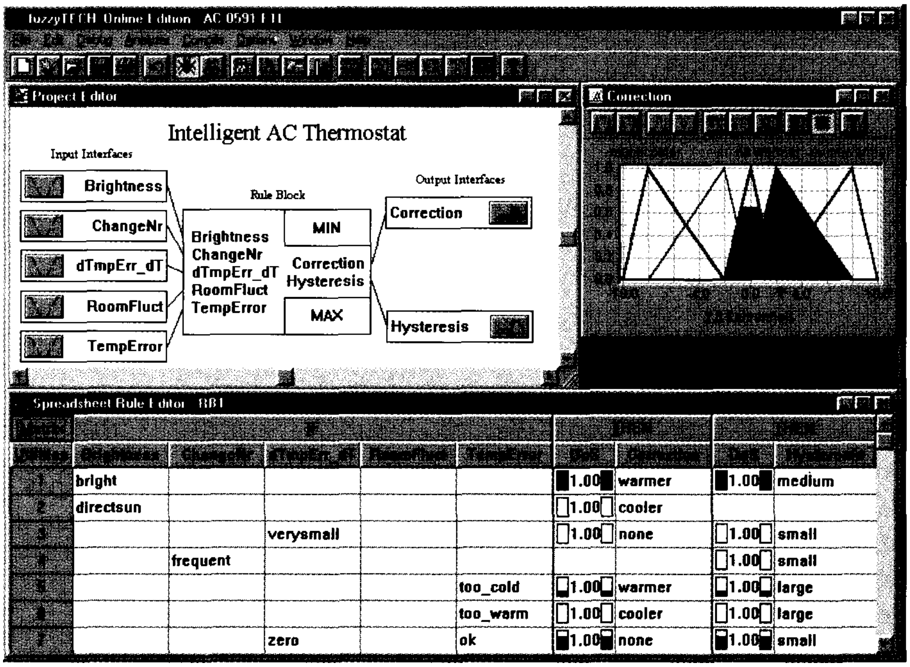


Photo 1—The *fuzzy-logic controller was designed in the* fuzzy TECH *software* system. *The upper left window shows the structure of the system, involving five input interfaces, one rule block, and two output interfaces. The window in the upper right corner shows the defuzzification of one output variable. The lower window shows part of the fuzzy rule base in* spreadsheet form.

The fuzzy-logic thermostat doesn't require any modification of the AC itself. By replacing existing temperature controllers, even old ACs can be upgraded. If ventilation is also controlled, even better performance can be reached in a more sophisticated design.

RAISING MIQ

This case study exemplifies how the application of fuzzy logic enables embedded intelligence in existing products. Even with products considered too mature for radical enhancement, integrating human experience and implementing experimental results can deliver significant improvements [4].

Professor Zadeh, the founder of fuzzy logic, calls this "raising the Machine Intelligence Quotient (MIQ)." Rethinking what systems and appliances should and could do for the user can create radically new products.

Because making machines respond to conditions in our lifestyle and usage patterns mostly involves human intuition and expertise rather than mathematical relations, fuzzy logic is an empowering technique for such designs. ❑

Constantin von Altrock began research on fuzzy logic with Hewlett-Packard in 1984. In 1989, he founded and still manages the Fuzzy Technologies Division of Inform Software, a market leader in fuzzy-logic development tools and turn-key applications. You may reach Constantin at cva@inform-ac.com.

REFERENCES

[1] S. D'Souza and C. von Altrock, "Fuzzy Logic in Appliances," Embedded Systems Conference, Santa Clara, CA, 1995.

[2] C. von Altrock, ***Fuzzy Logic and NeuroFuzzy Applications Explained,*** Prentice Hall, Englewood Cliffs, NJ, 1995.

[3] Inform Software Corp., ***fuzzyTECH User's Manual, 1996.***

[4] C. Okey et al., "Fuzzy Logic Controls for the EPRI Microwave Clothes Dryer," Third IEEE International Conference on Fuzzy Systems, Orlando, FL, 1348-1353, 1994.

SOURCES

PIC16C71
Microchip Technology
2355 W. Chandler Blvd.
Chandler, AZ 85224-6199
(602) 786-7200
Fax: (602) 899-9210
http://www.microchip.com/

***fuzzy*TECH**
Inform Software
2001 Midwest Rd.
Oak Brook, IL 60521
(630) 268-7550
Fax: (630) 268-7554
http://www.inform-ac.com/

IRS

407 Very Useful
408 Moderately Useful
409 Not Useful

#114

FEATURE ARTICLE

Designing Medical Electronic-Device Prototypes

Part 2: Testing for Electrical Safety

In getting medical devices to the point where they can be declared safe, there are specific tests that must be run. David gives us some step-by-step insight into what should be done and how.

Last month, I looked at designing electrically safe medical-device prototypes. I discussed the physical makeup of equipment, material usage, component selection, wiring, and PCB layout.

However, construction standards are only one aspect of applicable safety standards. Performance requirements are the other major concern.

Performance standards specify which tests apply to equipment and detail compliance criteria. Most construction requirements link to the performance details verifying acceptability.

The electrical tests probe insulation, components, and construction features which could cause safety hazards in normal or single-fault conditions.

In this article, I review safety-testing methods established by electrical safety standards for medical equipment. I present some useful test instruments that execute the test protocols.

GROUND INTEGRITY

The device's enclosure is the first barrier against electric shock. So, the first test assesses the integrity of the protective ground on a metallic enclosure and other grounded exposed parts.

UL standard 2601-1 establishes that impedance should be less than 0.1 Ω between the power plug's protective ground pin and each accessible part with the potential to become live if basic insulation fails. The standard requires that the test be conducted by applying a 50- or 60-Hz AC current with an RMS value of 10–25 A for 5 s.

But, you get a reasonable approximation of this test by using the 1-A DC current of the circuit in Figure 1. Resistance is assessed by measuring the voltage across the grounding path.

Op-amp U2 and power FET Q1 form a voltage-to-current converter driven by a reference voltage set by R6 to maintain a 1-A constant current on a conductor connecting the ground terminal of J5 and connector J2. Power comes from three alkaline D cells that provide a maximum voltage compliance of -4.5 V.

Because full-range circuit operation is accomplished by driving the gate of Q1 well above its source-to-drain voltage, U2 operates from 12 V generated by charge pump U1.

J2 and J3 connect to a Kelvin probe. This test probe separates the point where the current is introduced from the point where the voltage across the unknown resistance is measured.

This technique is required for low-resistance measurements because it effectively excludes the test leads' resistance. It also avoids voltage-measurement errors introduced by high-current-density concentrations on the current-injection terminals.

Convert a large alligator clip [e.g., Radio Shack 270-344) into a Kelvin probe. Replace the metallic axis with a nylon bolt that has nylon spacers isolating the jaws from each other. You also need to insulate the inner-spring ends. Separate leads connect to the probe's jaws-one to inject current and the other to sense voltage.

Once the circuit and probe are assembled, calibrate the adapter for exactly 1 A. Plug the instrument's power cord into a hospital-grade AC plug J1.

Then, clip the Kelvin probe to an exposed conductive point on the case.

The schematics are for illustrative purposes and are not intended to replace any standards. See page 28 of *INK* 74 for full disclaimer.

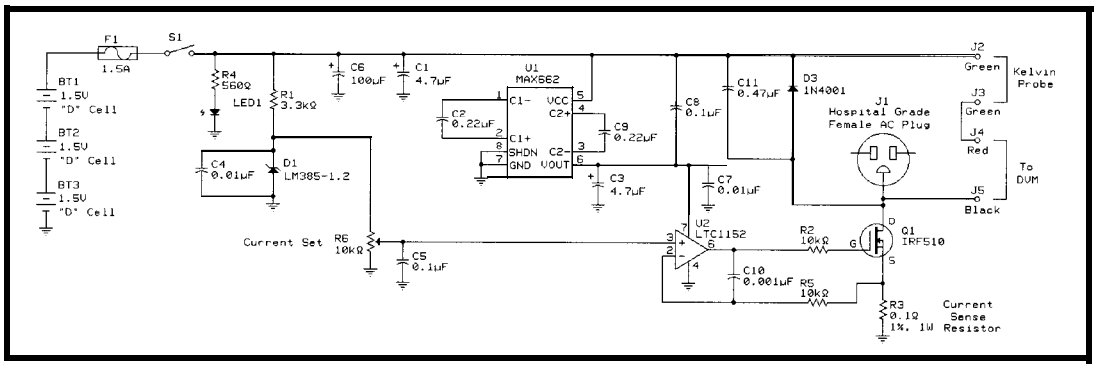


Figure 1--This simple adapter permits the measurement of milliohm resistances with any digital voltmeter. The circuit operates by generating a 1-A constant current on the unknown resistance of the ground path between the ground terminal of J1 and connector J2. A voltmeter connected across the power-cord ground conductor measures resistance in a scale of 1 V/.

A digital voltmeter connected between J4 and J5 directly reads the protective-ground resistance in a scale of 1 V/Q.

Remember the instrument's resistance measurement only approximates the standards' impedance test. The discrepancy is especially evident for high-power circuits, since a DC measurement of resistance doesn't convey information about the inductive component of impedance.

Moreover, DC ohmmeters are usually fooled by the polarized interface that results when an oxidation layer forms between defective ground-system connections. You can alleviate this concern by rerunning the test with the current-injection polarity reversed. Suspect nonlinear polarization indicating oxidation if resistance values are not significantly similar.

Failing this test is an immediate showstopper. Before further testing, you *must* locate the faulty connection responsible for compromising the integrity of the protective ground.

MEASURING CURRENTS

Leakage- and auxiliary-current tests are crucial for establishing an instrument's electrical safety. They are also the tests most commonly failed during safety-approval submissions and the periodic tests hospitals conduct to ensure the safety of medical electronic devices throughout their service life.

With medical electronic instruments, leakage- and auxiliary-current measurements are taken using a load which simulates human-patient impedance. The so-called AAMI load [1] is a simple R-C network that presents an almost purely resistive impedance of 1 kΩ for frequencies up to 1 kHz.

As Figure 2a shows, this load constitutes the core of the current-measuring device. If a 1-µA current is forced through the AAMI load at different frequencies, the measuring device's high-impedance RMS voltmeter gives the values of Figure 2b.

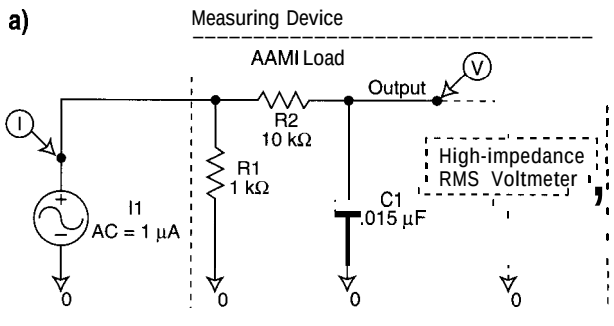
The frequency-response characteristics of the AAMI load are selected to approximate the inverse of the risk-current curve as a function of frequency. In turn, this curve is derived from strength and frequency data for perceptible and lethal currents.

Within 1-100 kHz, the current necessary to pose the same risk proportionally increases to 100 times the risk current between DC and 1 kHz. Since insufficient data exists above 100 kHz, AAMI decided not to extrapolate beyond 100 kHz. Instead, AAMI maintains the risk-current level of 100 kHz.

Current measurements should be conducted after preconditioning the device in a humidity cabinet. Access covers that can be removed without a tool must be detached. Humidity-sensitive components not contributing significantly to the risk of electrocution may be removed.

The equipment is placed in a humidity cabinet containing air with a relative humidity of 91-95% and a temperature t within the 2032°C range for 48 h (7 days if the instrument is to be drip- or splash-proof). Before placing it in the humidity cabinet, the equipment must be warmed to a temperature between t and t+4°C.

The measurement is done 1 h after the end of the humidity preconditioning treatment. Throughout this waiting period and during the testing, the same t must be maintained, but the relative humidity of the environment must only be 45-65 %.



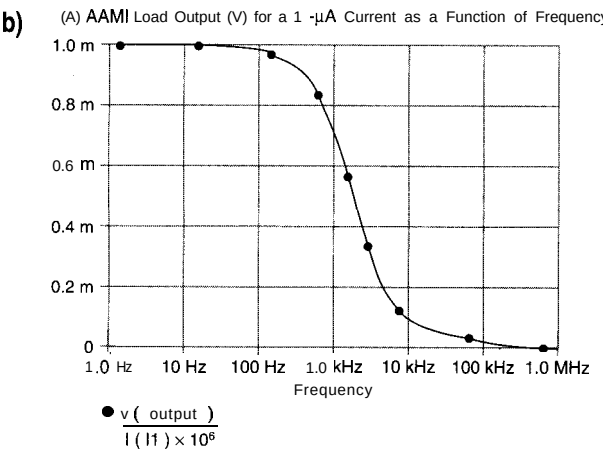


Figure 2—Leakage and auxiliary currents are measured using a load simulating the impedance of a human patient. a--The so-called AAMI load is the core of the current-measuring device defined by safety standards harmonized with EN-601-i. b—This load presents an impedance of approximately 1 k for frequencies up to 1 kHz.

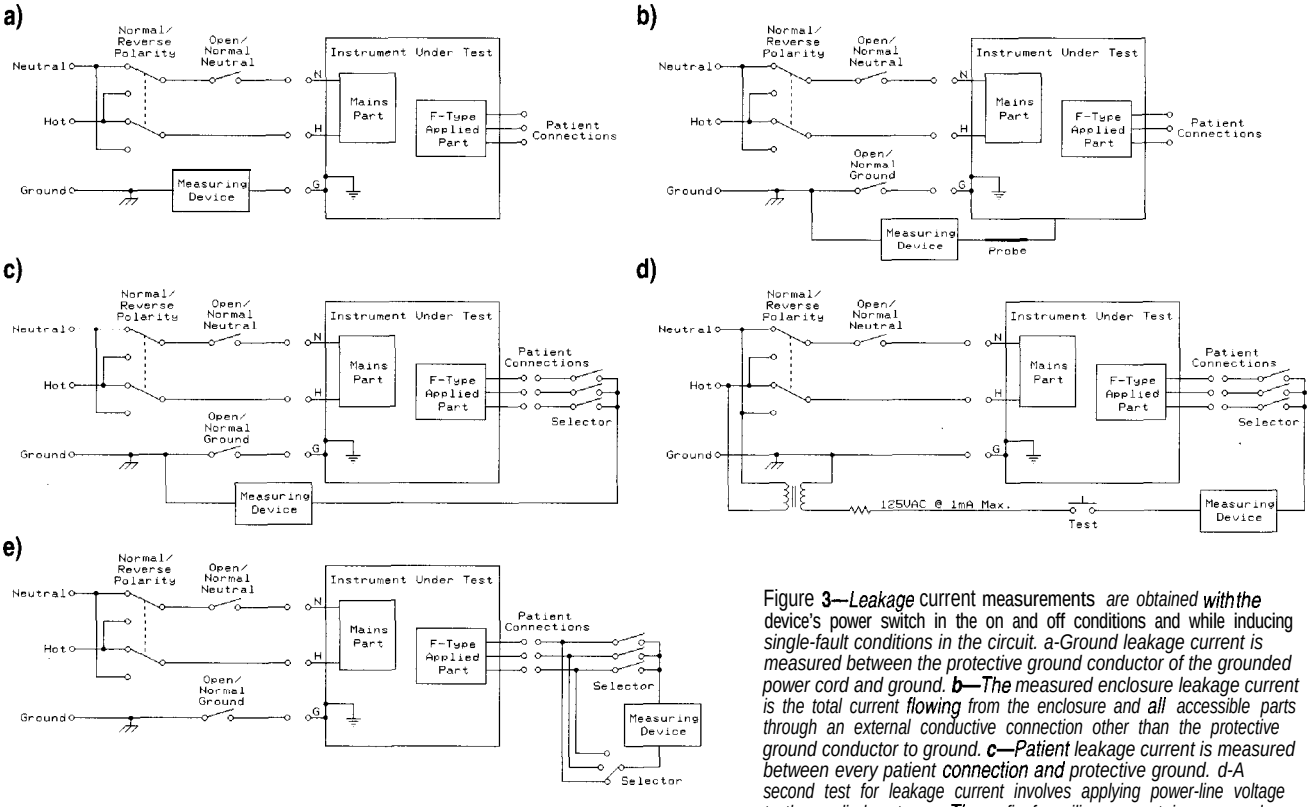


Figure 3—*Leakage* current measurements *are obtained with the* device's power switch in the on and off conditions and while inducing *single-fault conditions in the circuit. a-Ground leakage current is measured between the protective ground conductor of the grounded power cord and ground. b—The measured enclosure leakage current is the total current flowing from the enclosure and all accessible parts through an external conductive connection other than the protective ground conductor to ground. c—Patient leakage current is measured between every patient connection and protective ground. d-A second test for leakage current involves applying power-line voltage to the applied parts. e—The pafienf auxiliary current is measured between every patient connection and all other patient connections.*

Testing takes place with the equipment both on and off while connected to a power supply set at 110% of the maximum-rated supply voltage. When operational, the maximum-rated load must be used.

As mentioned in Part 1, allowable patient leakage and auxiliary currents are defined for normal and single-fault conditions. Single-fault conditions occur when a single means of protection against an equipment safety hazard is defective or a single external abnormal condition is present.

Certain single-fault conditions must be simulated during testing. They include the interruption of the supply by opening the neutral conductor and interruption of the protective ground conductor.

Patient leakage current between an F-type applied part and ground assumes an external voltage equal to 110% of the maximum-rated supply voltage is in direct connection with the applied part. For battery-powered equipment, the external voltage assumed to be connected to the F-type applied part is 250 V.

Leakage-current tests are conducted as shown in Figures 3a–d, with the device's power switch in the on and off conditions, creating the single-fault conditions specified in the figures.

If the enclosure or a part of it is made of insulating material, metal foil (20 x 10 cm] must be applied to the nonconductive part of the enclosure to protectively ground the enclosure connection. The foil is wrapped on the insulating enclosure's surface, simulating the way a human hand could act as a capacitively-coupled electrode.

Connections for measuring patient auxiliary currents are shown in Figure 3e. The current flowing between each patient connection and every other patient connection is measured under normal and single-fault conditions.

For this test, the measuring instrument should differentiate the AC from DC components of the RMS current reading. Different AC and DC auxiliary-current levels are permitted to flow through the patient (see Table 2 of Part 1, INK 74).

A VERSATILE MICROAMMETER

Figures 4-7 are schematics of an instrument that measures leakage and auxiliary currents. The circuit's core is an AAMI load converting a leakage or auxiliary current into a voltage waveform with a factor of 1 V/mA for frequencies under 1 kHz.

The AAMI-load output is amplified with a gain of 10 by instrumentation amplifier U1. The RMS value of the amplified signal is computed by U2, a true-RMS-to-DC converter IC, and displayed by a 3½-digit DVM module (e.g., Jameco 16029).

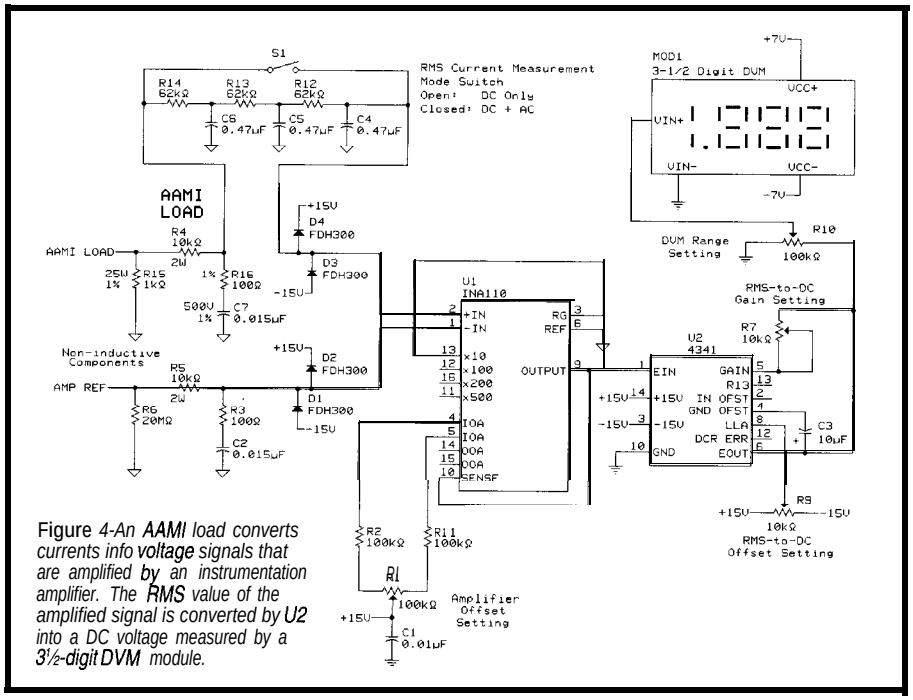


Figure 4-An AAMI load converts currents into voltage signals that are amplified by an instrumentation amplifier. The RMS value of the amplified signal is converted by U2 into a DC voltage measured by a 3½-digit DVM module.

R1 must produce a zero reading with no current flowing through the AAMI load. Potentiometers R7 and R9 should produce a reading of 1999 counts on the DVM for a 1.999-mA DC current through the AAMI load.

An R-C low-pass filter network can be interposed between the AAMI load and the voltage-amplifying circuit. This lets the DC component of a patient auxiliary current be measured.

The circuit in Figure 5 lets you easily configure the measurement setup to conduct the various leakage- and auxiliary-current tests required by the safety standards. Switch S3 selects the connection of the AAMI load to measure a patient current, the current on the protective ground pin, or the enclosure leakage current.

Switch S4 selects the patient-current source from the different possible combinations of patient connections. Through this circuit, power-line-level voltages can be applied to the patient connections by way of relay K1.

Figure 5 shows the connection distribution suitable for testing a 12-lead ECG. However, any other applied part's connections can be substituted.

The connectors establishing connection to an applied part's leads must be carefully selected so they don't contribute to the measured leakage or auxiliary currents. Ohmic Instruments' 301PB ECG binding posts fit the standard snap-ons or the pin-tips used for connecting to ECG patient electrodes.

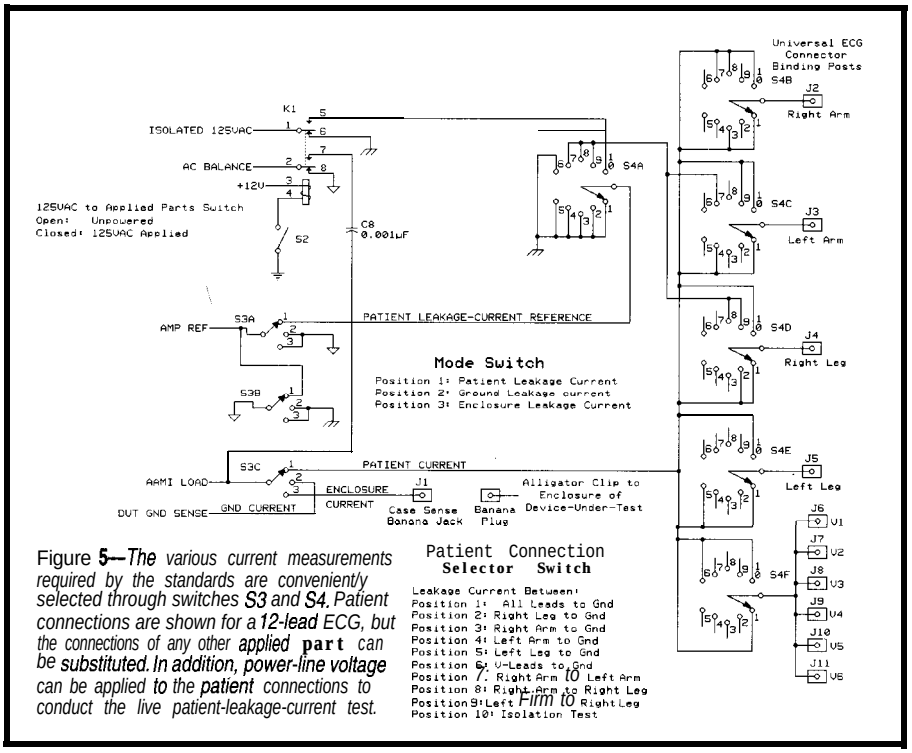


Figure 5—*The various current measurements required by the standards are conveniently selected through switches S3 and S4. Patient connections are shown for a 12-lead ECG, but the connections of any other applied part can be substituted. In addition, power-line voltage can be applied to the patient connections to conduct the live patient-leakage-current test.*

The circuit in Figure 6 controls the power supplied through J13 to the device being tested. SPDT relays with contacts rated for 125 VAC at 20 A reverse the power-line polarity at J13 and cause open-ground and open-neutral single-fault conditions.

Three neon lamps indicate that the measurement instrument is powered and verify that the AC plug providing power is correctly wired. Normal and fault conditions are shown in Table 1.

Figure 6 also shows the 115-VAC isolation transformers and the voltage divider network formed by R17–R19 which generates the 125 VAC for measuring the patient leakage current with applied powerline voltage. R20 limits the current delivered by this circuit to approximately 1 mA.

Figure 7 presents the DC power-supply section of the measurement instrument. Linear regulators generate the various power buses required by the ammeter. In addition, R24–R26 derive an AC signal to balance the measurement circuit while applying 125 VAC to the patient connections.

When measuring currents, place the equipment being tested on a nonconductive bench away from grounded metal surfaces. Ensure all external areas of an applied part, including patient cords, are placed on a dielectric insulating stand (e.g., a polystyrene box] approximately 2 m above a grounded metal surface.

A word of caution: be extremely careful when using this instrument! Unrestricted power-line voltages power the device being tested, making the risk of electrocution or fire very real.

Single-fault conditions forced during testing may result in the device's enclosure becoming live, threatening anyone who accidentally touches it. Since power-line-level voltages can be injected into patient connections and the associated power system, never conduct these tests in the vicinity of a patient or on a power-system branch that powers medical electronic instruments connected to patients.

Indicator Lights LP1	LP2	LP3	Condition
Off	Off	Off	Instrumerit off, or open HOT
Off	On	Off	Open NEUTRAL
Off	On	On	HOT/GROUND reversed
On	Off	Off	Open GROUND
On	Off	On	HOT/NEUTRAL reversed
On	On	Off	Correct, or GROUND/NEUTRAL reversed

Table *I--Three neon lamps used in the circuit in Figure 6 verify that fhe AC plug from which power is obtained is correct/y wired. Normal and fault conditions are represented by combinations of the lamps.*

THE **HIPOT** TEST

Once compliance with current leakage limits is established, high-potential (HiPot) application testing assesses the suitability of the insulation barriers between an instrument's isolated parts.

Very high voltage is applied differentially between the parts separated by the isolation barrier. As shown in Table 2, the test voltage is dependent on the voltage *U* to which the barrier is subjected under normal operating conditions at the rated supply voltage.

While high voltage is applied, the current is monitored to ensure that no arc breakdown occurs. HiPot testers have internal circuitry which automatically disconnects the high-voltage supply across the insulation when current exceeds a preset threshold.

Although the standards allow slight corona discharges, excessive RMS leakage-current measurement does not reliably detect dielectric breakdown. Monitor milliamp-level current spikes or pulses. These indicate the type of arc breakdown occuring prior to catastrophic and destructive breakdown.

Again, the voltage should be within the rated operating frequency of the tested instrument. Measured magnitudes refer to their AC RMS values.

Despite this, the technique is sometimes modified by applying the DC equivalent to the peak-to-peak amplitude of the required AC RMS voltage. Since capacitive and inductive coupling disappear, this change reduces current leakage between parts, but leaves a current signal which directly conveys information about insulation-breakdown processes at the peak of the dielectric stress.

So, if an instrument's highest-rated supply voltage is 125 VAC, the standard requires that basic insulation tests be conducted at 1000 V_{RMS}. The peak-to-peak voltage of the required AC test signal is:

$$1000\ V_{RMS} \times 1.41 = 1410\ V_{p-p}$$

As such, 1410 VDC is applied between a wire connecting hot and neutral power-cord connections together and another wire attached to the protective ground connection of the instrument.

Similarly, the insulation barrier between an F-type applied part and any other point of the instrument must be tested at 3000 V_{RMS}. This corresponds to a 4230-VDC voltage for the modified test. Insulation breakdown is indicated by current spikes appearing when the high voltage is applied between a point tying all patient connections and a point tying all nonisolated I/O lines and the hot, neutral, and ground connections of the power cord.

Although convenient, this method is not always accepted by regulatory

bodies as a reasonable substitute for the tests specified by the standards. In any case, make sure that the HiPot tester you use can detect precatastrophic breakdowns. Otherwise, your instrument's insulation may break down and instantaneously deliver a lethal level of current.

HiPot testing should be conducted after humidity preconditioning. As before, all access covers that can be removed without a tool must be detached. Humidity-sensitive components not contributing to the risk of electrocution may be removed.

Voltage-limiting devices (e.g., spark-gap transient-voltage suppressors, Iso-Switches, etc.) in parallel with the insulation to be tested can also be removed if the test voltage would make them operative during the HiPot.

HiPot tests of the various insulations must be conducted with the instrument still in the humidity cabinet. For each test, voltage should be slowly increased from zero to the target potential over a 10-s period and then kept at the required test level for 1 min. If breakdown does not occur, tripping the HiPot tester's automatic shut-off, the test is completed by lowering the voltage to zero over 10 s.

The standards do not exclude battery-powered equipment from HiPot testing. Instead, the reference voltage U is set to 250 V.

Fully or partially nonconductive enclosures are also tested. Use the same metal-foil method as in current-leakage testing, being careful that flashover does not occur at the edges of the foil at very high HiPot-test levels.

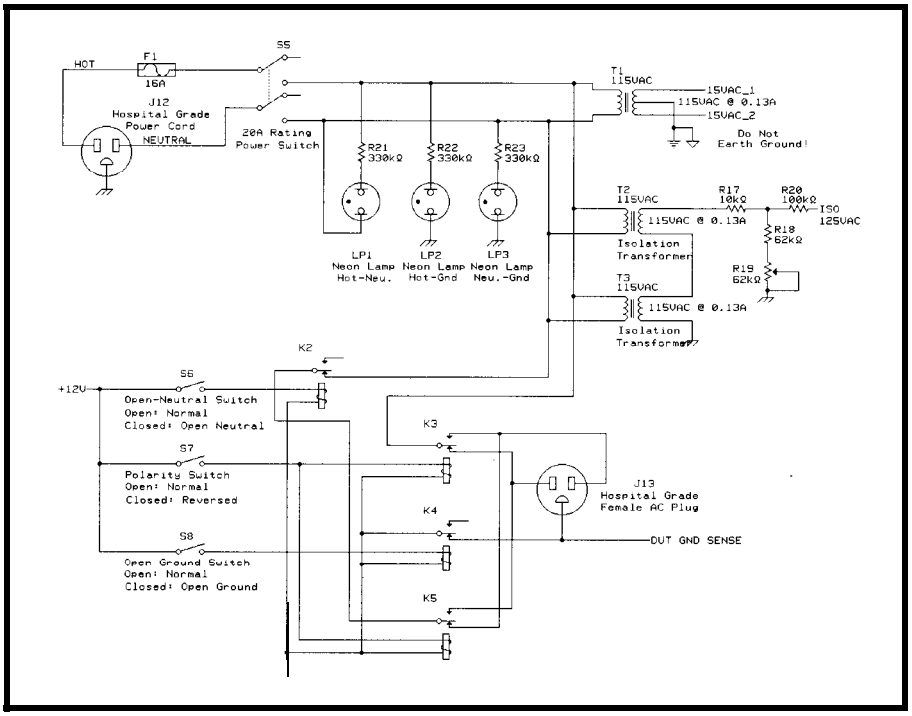


Figure 6—*Power-control* relays reverse power-line polarity and cause single-fault conditions for analyzing *the instrument* under *test.* Two *115-VAC isolation* transformers and *the* voltage-divider network formed by *R17–R19* are used *to* generate *125 VAC for* measuring *the* patient leakage current *with applied power-line voltage.*

TESTING FOR OTHER RISKS

You'll probably get by successfully passing the tests outlined above if the prototype's evaluation is conducted on a very limited number of patients and under a physician's supervision.

If the instrument is built solidly enough to inspire confidence, the engineering evaluation prototype is not usually required to pass the battery of tests to ensure compliance with the mechanical and labeling requirements demanded for prerelease or commercial products.

However, other potential risks exist in a clinical environment. You should make sure that you will not cause undue interference or harm through other mechanisms besides leakage currents.

The equipment must minimize the risk of fire and explosion. Safety standards typically limit temperature rises and define enclosure requirements to contain fires within the instrument. If the device operates where flammable anesthetics or oxygen-enriched atmospheres are (e.g., operating and hospital rooms), special requirements ensure explosive atmospheres are not ignited.

If intentional sources of ionizing radiation are present, the equipment must be evaluated by the Center for Devices and Radiological Health. If components may generate ionizing radiation not used for a diagnostic or therapeutic purpose (e.g., from CRTs), you must ensure exposure at 5 cm from any accessible part of the equipment and averaged over a 10-cm^2 area is less than 0.5 mR per hour.

Devices using ultraviolet radiation and lasers are also regulated in specific substandards of the IEC601-2 series.

The equipment must not emit EM1 so other equipment malfunctions. It should minimize conducted emissions to the 0.15–30-MHz range and radiated emissions to 150 kHz to 1 GHz.

At the very least, a near-field survey of the equipment should be conducted to identify potentially offending emissions (see "Sniffing EM1 in the Near Field," INK 71). Scan results should be passed on to the host hospital's biomedical engineering department.

Parts that contact a patient's biological tissues, cells, or body fluids must be assessed for biocompatibility. Be careful how you select materials. Test for biological effects, including cytotoxicity, sensitization, irritation, and intracutaneous reactivity.

The equipment must have a way to be immediately and completely disconnected from the patient in case of an emergency. Connectors used on

Insulation Type	u 50	50 < U 150	150 < U 250	250 < U 1000	1000 < u 10000
Basic	500	1000	1500	2U+1000	U+2000
Supplementary	500	2000	2500	2U+2000	U+3000
Reinforced	500	3000	4000	2(2U+1500)	2(2U+2500)

Table 2—*Insulation* barriers *are HiPot* tested at various *voltages.*

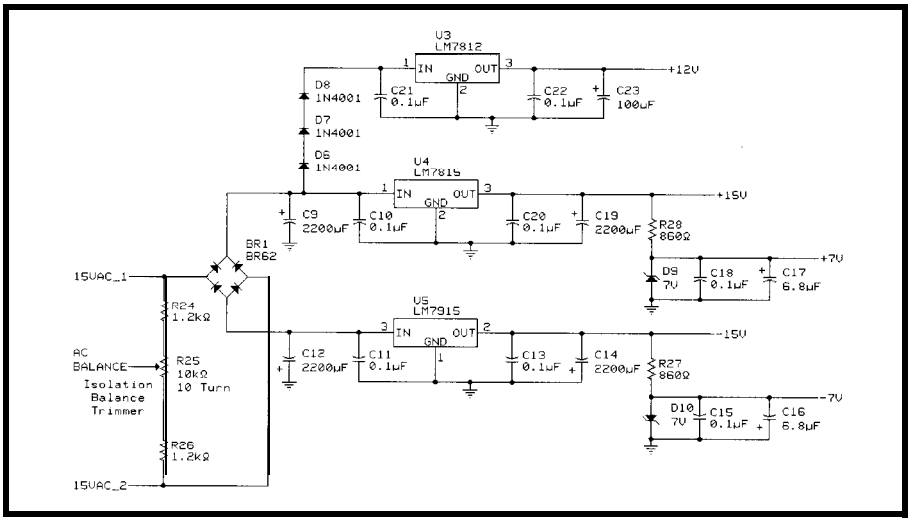


Figure 7—*Linear regulators generate the various voltages required by the ammeter's circuitry. In addition, R24–R26 derive an AC signal to balance the measurement circuit during application of 125 VAC to the patient connections.*

patient connections should be clearly and uniquely identified. They must be of a type which cannot be accidentally plugged into the power line or form an electrical path to any point when they are disconnected from the equipment.

Educate clinical staff about your instrument's potential dangers, regardless of how remote and unlikely they may be. If something goes wrong, they have to save the patient's life!

BEYOND PROTOTYPING

You may be wondering how to turn your prototype into a commercial product. First, conduct preliminary safety and functional benchtesting.

Invest a little time researching possible patent infringement. Then, show your prototype to a physician who may help evaluate your idea by testing the device on animals and human subjects.

If all goes well and testing shows your idea truly marks an advancement to medicine, you may be tempted to invest in the infrastructure, tools, and materials required to mass-produce and sell your device. But...*don't*!

Unlike many other high-tech businesses, the medical-device industry is highly regulated. Especially in the U.S., the level of regulation is so extreme that compliance concerns often outweigh all other technical and financial considerations.

Even if you built your device at costs you could cover out of your own pocket, the extensive clinical trials and lengthy submission process needed to satisfy the FDA would probably make a personally-based venture completely unviable. It's not uncommon for the costs of a bare-bones clinical trial and submission for a simple device to reach several million!

The FDA reasons that these regulations ensure the safety and efficacy of a new medical instrument before it's used in the general population.

I don't mean to discourage you from pursuing your idea. But, be aware of the unique characteristics of biomedical startup.

MARKET PULSE

Applying the safety standards' principles and requirements is important for even the first engineering evaluation prototype of a medical electronic device. The biomedical-equipment department of any hospital hosting the preclinical trials will demand that the device passes all electrical safety tests at a bare minimum.

And, since we live in a litigious society, it's a good idea to maintain clear records showing you carefully considered the standards and regulations from the beginning.

I'm sure you can appreciate from this overview that there are many stringent safety and performance requirements for medical devices. However, requirements are not enforced by to discourage medical advancements.

It's my perception that applicable standards and regulations help the designer develop a product which provides true benefit to the patient while reducing foreseeable risks.

Pursue data that clearly demonstrates clinical efficacy for any ideas you have for a medical product. And, realize that to be approved, new medical technology ***must*** be based on solid physiological and technical grounds.

Armed with this information-and if you can adapt to a changing regulatory environment-I'm convinced a receptive audience of investors awaits your idea. ❑

David Prutchi has a Ph.D. in Biomedical Engineering from Tel-Aviv University. He is an engineering specialist at Intermedics, and his main R&D interest is biomedical signal processing in implantable devices. He may be reached at davidp@mails.imed.com.

REFERENCES

[1] Association for the Advancement of Medical Instrumentation, ***Safe Current Limits for Electromedical Apparatus,*** ANSI/AAMI Standard ES1, 1993.

SOURCES

INA110 instumentation amplifier IC, 4341 RMS-to-DC IC
Burr-Brown Corp.
6730 S. Tucson Blvd.
Tucson, AZ 85706
(520) 746-l 111
Fax: (502) 889-1510

MAX187
Maxim Integrated Products, Inc.
120 San Gabriel Dr.
Sunnyvale, CA 94086
(408) 737-7600
Fax: (408) 737-7194

301PB ECG binding posts
Ohmic Instruments Co.
508 August St.
Easton, MD 21601
(410) 820-5111
Fax: (410) 822-9633

I R S

410 Very Useful
411 Moderately Useful
412 Not Useful

FEATURE ARTICLE

David Rector

Getting Started with Xilinx EPLDs

Part 2: Hands-On Project—Concept and Design

This article picks up where Conrad Hubert left off in his introduction of EPLDs last month. David brings EPLDs down to earth by giving us hands-on programming tips. Listen up for details.

Computer hardware has revolutionized the scientific world as much as the oscillograph did almost 100 years ago. In my research, it has helped me visualize brain activity using computer imaging technology.

Accurate timing of computer-controlled events is critical for electrophysiological data. So, I developed a timer/counter board with microsecond precision.

In Part 1 (*INK* 74), you saw a simple application with a tutorial on programming Xilinx EPLDs. In the next two parts of this series, I'll develop a more complex design and implementation using schematic-capture software. I'll also describe a circuit that plugs into an ISA slot of an IBM-compatible computer.

I designed the circuit with the following specifications:

- to read the full count (i.e., 36 h, 6 bytes) without a change in the counter value during the readout, all values register on command.
- to prevent the registers from latching during a count transition, the latch command holds until the count transition completes.
- to synchronize computer events with other equipment in experiments (e.g,. polygraph strip-chart records), computer time must print in a BCD-encoded analog form onto paper.

Such procedures are common in scientific studies using a device called the AIL-code generator, developed by Airborne Instrumentation Laboratories.

Figure 1 illustrates a sample AIL-code waveform. Every 10 s, the current time is written in a BCD-encoded format beginning with the most significant digit of the seconds counter.

The waveform is interpreted as a series of bumps-small bumps representing a 0 and tall bumps representing a 1.

The circuit producing this waveform consists of nine main components (see Figure 2) and two EPLDs. The ISA interface is entirely contained on AILISA, a small EPLD chip.

AILISA communicates between the AIL-code generator and the host computer, and it controls external I/O and programming the waveform SRAM. A larger EPLD (AIL chip) generates the counts and produces the waveform.

At the core of the AIL chip are two counters. A 20-bit binary counter which resets at 1,000,000 makes up the microsecond counter, and a 20-bit, S-digit decade counter counts from 0 to 99,999 s.

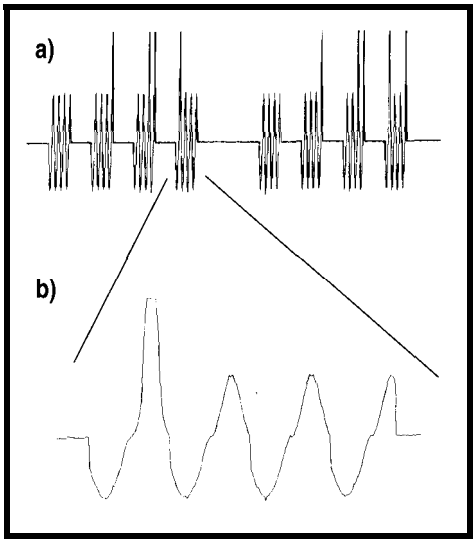


Figure 1a--The top *par! of this figure illustrates a 20-s clip of the analog waveform output.* ***b—****Two BCD-encoded seconds counts are* shown. *One of the digits is expanded. The values (read left to right) are interpreted as a counter status of 1380 s and 1390 s* since the *start* of *the* count sequence.

The 40-bit latch holds the current counter status after a request from the latch-request circuit. A 4-to-1 mux selects one bank of 16 bits from the 40-bit latch to be transferred to the host computer. Either 16 or 8 bits are read out by multiplexing the data lines.

The counter outputs also drive the SRAM address generator which produces the desired waveform through a DAC.

The circuit's concept is simple. This circuit can produce any waveform with a 0.25-s period. With small changes, it can make any kind of waveform.

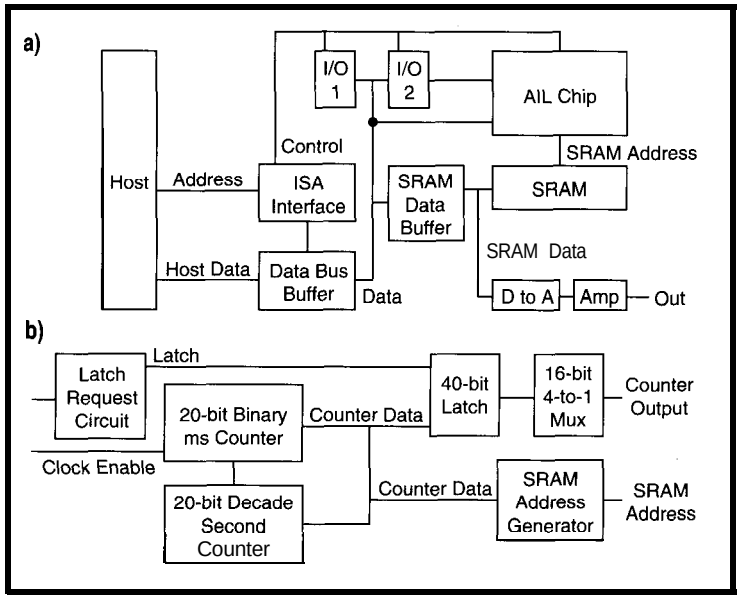


Figure 2a—*The AIL generator* circuit includes the *host computer, ISA glue logic, I/O register blocks, SRAM waveform storage, a DAC, and the AIL-chip circuit* **b**—*Inside the AIL chip are the microseconds and seconds counters, a 40-bit latch, latch-request circuitry, a 16-bit 4-to-1 multiplexer, and an SRAM address generator to produce the AIL* waveform.

INITIAL DESIGN STEPS

To give you an idea about the advantages of EPLD technology, the first application of this waveform generator was constructed from discrete gate components. It required 37 separate ICs and a full-length ISA prototyping card. It cost about $100.

The current implementation requires two EPLD components, 12 integrated circuits, and a half-length ISA card and costs about $100 as well.

Until the price of EPLDs drops, the cost looks the same, but the EPLD's advantages far outweigh discrete components. The smaller size fits better on the new compact motherboards. And, far less wiring means less construction time and fewer wiring mistakes.

I gain more design flexibility from the unlimited availability of gate functions. Also, virtual design methods let me design and test before I solder.

Finally, with reprogrammable EPLDs, mistakes are easy to fix. Since I started using them, I rarely use discrete components. In the future, I think discrete components will be as rare as an individual transistor.

VIRTUAL CIRCUIT

To get started, you need at least one of the Xilinx software packages. When you begin designing with EPLDs, there are two directions to take.

The cheapest and simplest method involves describing each output of the chip as a Boolean equation of the inputs. For complex designs such as this one-there are 105 equations-the Boolean description is tedious at best.

However, the cost investment is minimal. The Xilinx tools that create the programming file (DS-550) are $89, and the Deus Ex Machina programmer is $295. I also needed an additional adapter board, the XPGMA7, which cost about $145.

For this design, however, I used Xilinx's ViewLogic stand-alone system, DSVLS. Although it costs about $2500, it's worth the extra money if you do a significant amount of EPLD or FPGA work. It includes timing simulation and one year of technical support, as well as online documentation and design tutorials.

Xilinx's XACT 6.0-DSVL software, which costs $995, is part of the DSVLS package and includes a wide variety of function macros which ease the design process. At the end of schematic entry, XACT checks for errors, generates the Boolean equations, and prepares the timing simulations.

An infinite number of tricks are possible to optimize specific designs. Even though the EPLD tutorial and documentation from Xilinx are excellent, it's impossible to consider all the possibilities. That's why I mentioned that one year of technical support!

AIL CHIP

Now, you're ready to enter the schematic into the schematic-entry software. I used the ViewLogic system, but the Xilinx software interfaces to a number of schematic-capture programs (e.g., Mentor Graphics and OrCAD).

Figure 3 illustrates the schematic for the AIL chip EPLD fit into the XC73108-15PC84 which consists of several components provided as macros from Xilinx and three modules I created to simplify the drawing.

Two flip-flop macros, FDC and FDRE, represent the latch-request circuit. Two modules I created, USEC_CNT and SEC_CNT, generate the counts latched by 40 flip-flops, FD16RE, and FD4RE.

Another module I created is the M4_1L16, which selects 16 bits from the 40-bit latch to be read by the host computer.

Finally, the M16_1E is a multiplexer macro from Xilinx which operates as part of the SRAM address generator to select which seconds digit to display on the waveform.

Every input and output of the chip requires a buffer macro to bring the signal into or out of the chip. The IBUF symbol is always used for inputs unless the input performs a special function.

There are two BUFFOE input lines on most Xilinx packages, which drive OBUFEX, a special high-speed tristate output symbol.

Outputs must have either an OBUF, an OBUFE (i.e., noninverted enable), or an OBUFT (i.e., inverted enable) symbol which are tristate outputs, or the special OBUFEX symbol.

Also, the BUFG symbols are used to drive special fast clock lines reserved for this purpose. There are two fast clock lines on most Xilinx packages.

For a logic equation to be mapped into an FFB, all its clock lines must be driven by a fast clock line. The presence of the fast clock lines make it easy to design compact synchronous circuits.

COUNTERS

Figure 4 illustrates the microsecond and second synchronous counter modules.

The ViewLogic schematic-capture software contains utilities to create modules, simplifying your schematic's appearance. Each of the counters was created as a separate module which appears in the final schematic as a functional block.

Xilinx provides two series of macros for counters with various count lengths, both loadable and nonloadable, with clear and reset. The first series contains generic counters CB*xx* (binary, xx = bit length) and CD*xx* [decade), which are primarily used in FPGA designs.

In this design, the AIL code depends on seconds displayed in decimal format. It's easier for someone looking at the waveform to interpret the number. Even though Xilinx provides a decade counter macro CD4RE, I didn't use it. It isn't speed optimized for EPLDs, and it requires more resources to implement.

The series used CB16X2 and CB4X-2, which are optimized for EPLD implementation. I used binary counters for the seconds counter, then turned them into decade counters with AND gates.

Next, examine the counter module design. The battle between speed and density is ever present in EPLD designs. Lucky for us, Xilinx constructed EPLDs with two types of function blocks and a universal interconnect module (UIM).

The regular function block (FB) contains macrocells which are optimized for high logic density. The FFB is optimized for speed, but it handles less logic.

To optimize speed and density of your design, it's easier to use Boolean descriptions of your circuit and forget about schematic entry.

Otherwise, if you are constructing your design with a schematic-capture program, you'll definitely want to look at the equation files as you go along.

The Xilinx software reports how your fastest signal paths fit into the FFB, while monitoring how your complex equations are fit into the FBs. I often encounter situations where my equations get so complex that my FFBs contain no logic.

For complex equations, you must split your equations into smaller components. Always keep in mind that if the XACT fitting software cannot fit an equation into a single macrocell, it must break it up into multiple parts. Also, the fitter is not always very intelligent about where it splits the equations.

This application is a good example of a splitting phenomenon. The outputs of the seconds counter drive a 16-to- 1 multiplexer which all converge to MA20, a single output, which has over one hundred inputs. For optimum speed, this equation requires an entire function block.

Since speed isn't critical in this circuit, I can break up the seconds output by placing a B U F macro on each counter bit. This technique forces the equation to be split at this point.

The microseconds counter does not require B U F macros because each of the counter's output bits drives an output buffer and an output pad. The equation must be split at these points.

If you haven't worked with PLDs before and you're using a schematic-capture program to enter your design, remember that a flip-flop and a few gates are not necessarily represented physically as with discrete components. An output equation will be reduced by the fitting software to have only one latch and any number of combinatory inputs leading into it.

SYNCHRONICITY

The XACT programming package was my first exposure to synchronous design. After years of designing with discrete gates, it was a painful transition, and I made every possible mistake. I encountered every possible race condition as I struggled to stuff asynchronous paths into my EPLDs.

The AIL timer/counter chip eventually became entirely synchronous as I

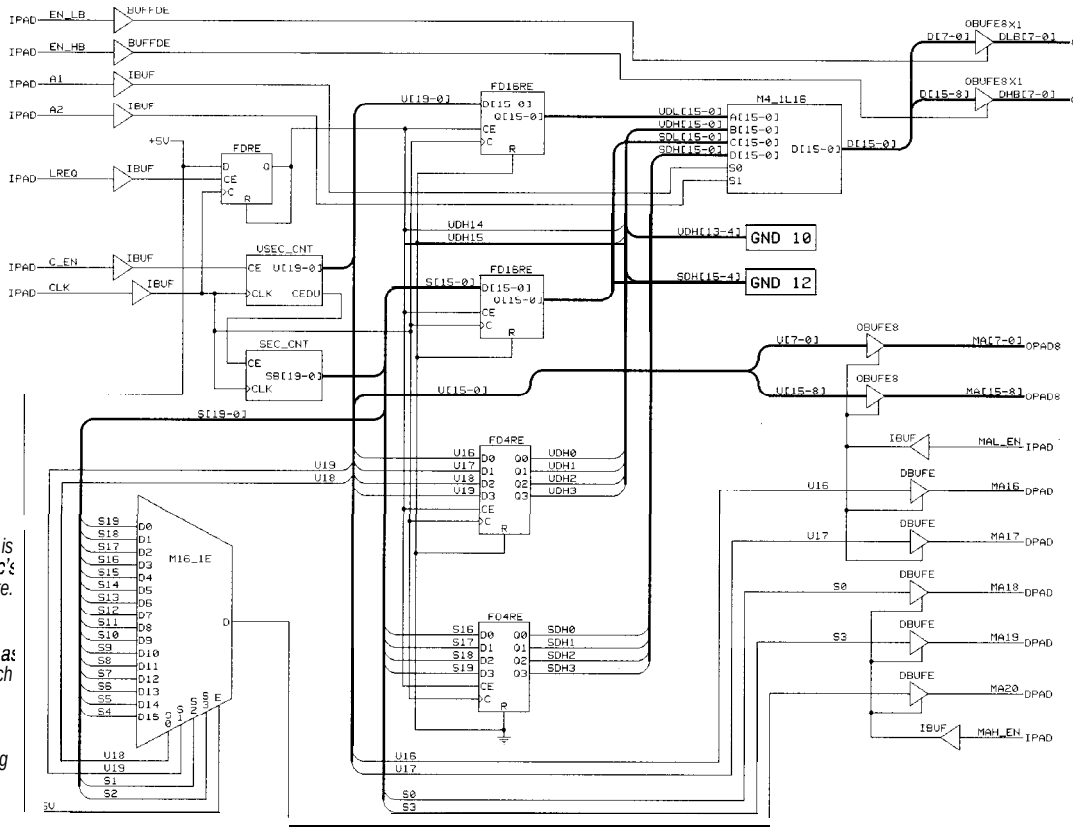


Figure 3—*The schematic is entered through ViewLogic's schematic-capture software. Xilinx* provides functional *macros such as flip-flops (FDC and FDRE) as well as complex logic symbols such as the 16-to-1 multiplexer M16_1E. Connect the symbols* with either sing/e net lines *(thin) representing sing/e wires or bus lines (thick) representing many* wires.

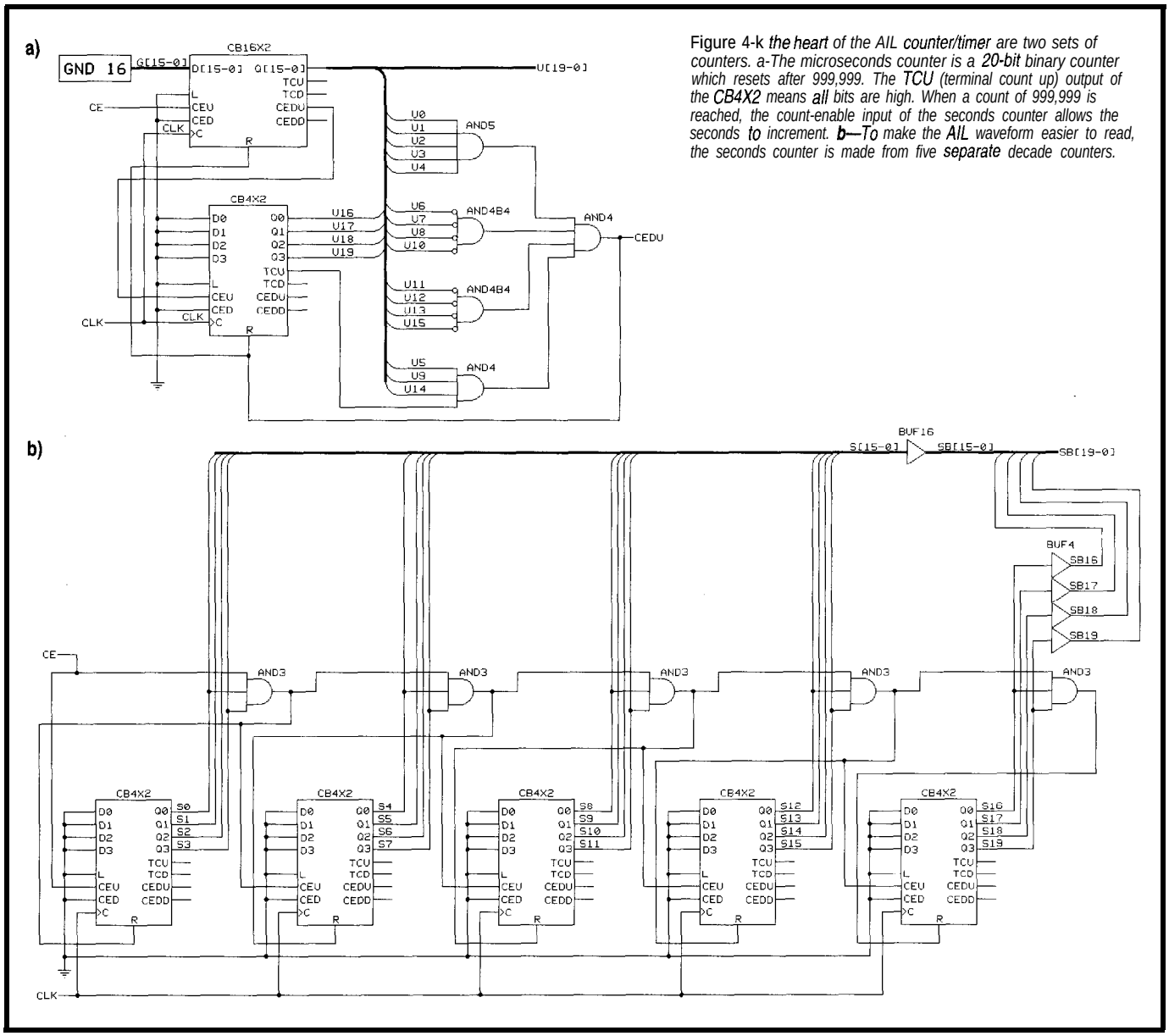


Figure 4-k *the heart of the AIL counter/timer are two sets of counters. a-The microseconds counter is a 20-bit binary counter which resets after 999,999. The TCU (terminal count up) output of the CB4X2 means all bits are high. When a count of 999,999 is reached, the count-enable input of the seconds counter allows the seconds to increment. b—To make the AIL waveform easier to read, the seconds counter is made from five separate decade counters.*

learned the tricks. Most importantly, you must have a latch on every output, then drive every latch on the chip with the same clock signal.

If you can do that, you have perfected the synchronous design technique. Here's a hint-always use the clock-enable inputs rather than drive the clock line of flip-flops.

As you'll see, my AILISA glue-logic chip is still asynchronous, but the blinding speed of the EPLD saved me. It would have been better if I latched the outputs with the ISA clock. It would also have been easier to implement a wait-state generator.

But, I took my discrete-component design and dropped it right into the EPLD without any problems. The guys at Xilinx may scold me for not designing a proper state machine for the AILISA glue logic, but time was running short.

LATCHES

After placing the counters, I connected each output bit to an FD16RE or FD4RE macro which latches the count state when the CE clock-enable line goes high. The Xilinx macro library contains hundreds of different flip-flop and latch macros, including toggle and S-R flip-flops. Each macro is optimized for its performance.

The flip-flops in the 40-bit latch register the counter state only when the latch request flip-flop FDRE is active. You need to hold the latch-request input LREQ high for at least one timer-clock pulse width.

The outputs of the latches feed into a 16-bit 4-to-1 multiplexer with independently selected tristate S-bit output buffers and pads. Download the details about what is in the M4_1L16 module from the Circuit Cellar BBS.

The multiplexer M4_1L16 module does most of the work in interfacing the registered counter states to the host computer.

Inside the M4_1L16 module are 16 separate 4-to-1 multiplexer macros, M4_1N. Four 16-bit data lines-UDL, UDH, SDL, SDH-represent the microsecond and second counter states in two 16-bit words each, selected by the A1 and A2 input lines.

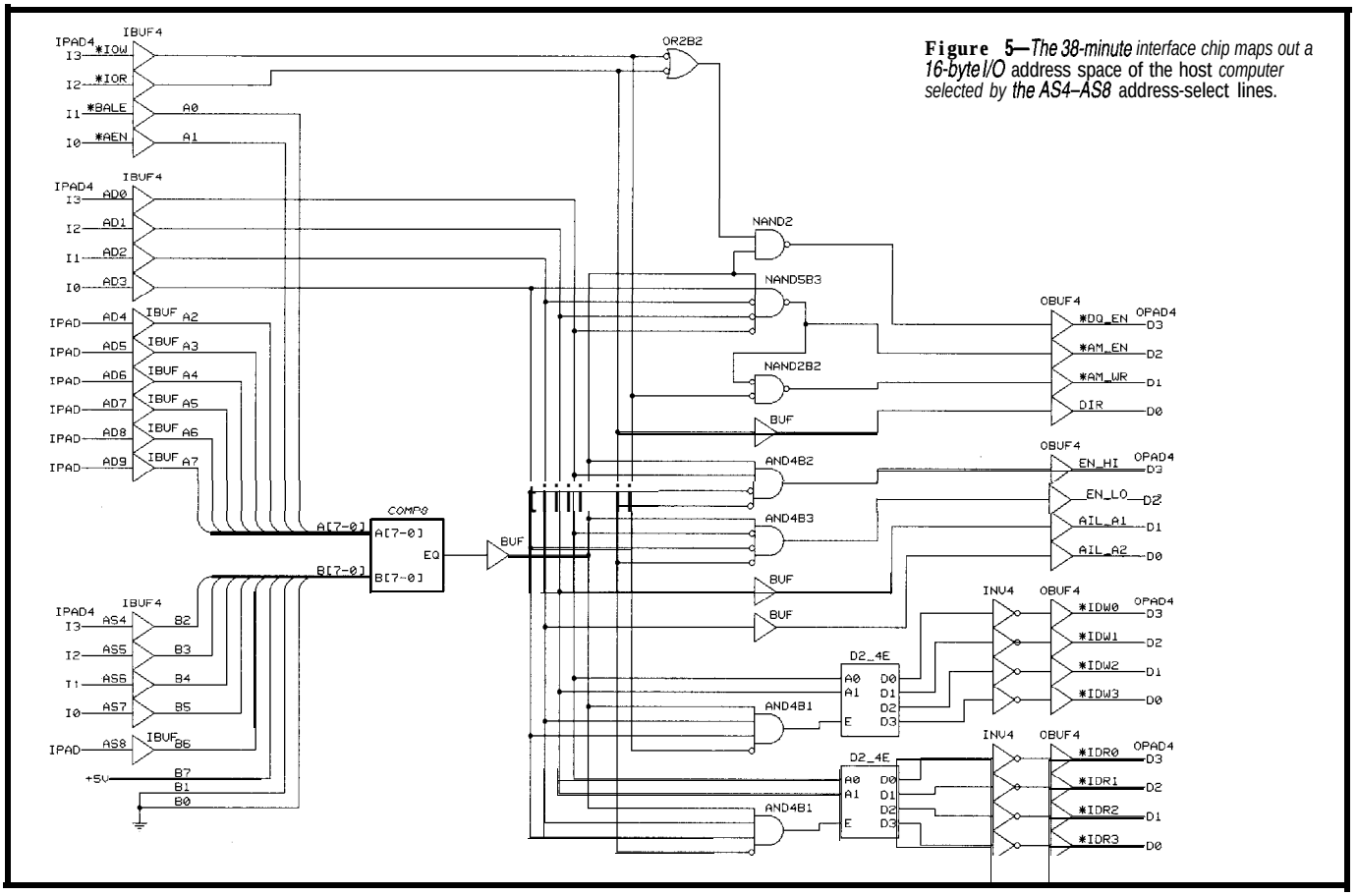


Figure 5—*The 38-minute interface chip maps out a 16-byte I/O address space of the host computer selected by the AS4–AS8 address-select lines.*

Using discrete components, I typically use tristate buffers to create the data-bus multiplexer. Within the EPLD, tristate outputs are precious resources because each one requires an output pad.

It may look more complicated in the schematic, but within the UIM, the multiplexers can be implemented with far fewer resources. I decided to create two independent 8-bit lines to allow the option to use an 8- or lh-bit bus.

To ensure that inactive lines are low, I drive unused inputs to the multiplexer with 22 ground symbols packaged in the GND10 and GND12 modules. This feature takes no additional resources because it is implemented in the UIM and included as part of the output equation.

The host computer reads UDH14 to make sure the chip is finished latching.

It can also be used to see if the chip is done with its reset sequence. During reset initiated by bringing the MR master-reset pin of the EPLD low, all output latches are held high.

SRAMADDRESSGENERATOR

The waveform generator is implemented by driving the addresses of an SRAM chip with the output bits of the counters. Output lines MAO-MA20 represent 21 bits of address space for the SRAM.

In this application, only the upper 13 bits are used. Depending on the resolution that you need for the waveform, you could use more address bits.

Address bits MAO-MA1 7 are equivalent to the microsecond-counter states UO-U17. Since the microsecond counter resets after 1,000,000 counts, it isn't a purely binary count. The last few counts in the binary sequence are missing, but this maintains ease for the host in interpreting the microsecond value.

The microsecond counter could also be implemented as a pure 20-bit counter that is driven with a 1.024-MHz clock. Each count would represent 0.98 us.

To keep my count interpretation simple, I ignored the lost counts after 1,000,000 in my waveform.

The counter state outputs U18, U19, S1, and S2 drive the 16-to-1 multiplexer M 16_1E which selects which seconds output line, S4 to S19, to write onto the waveform, MA20. SO and S3 select which part of the AIL waveform the chip currently displays.

38-MINUTE INTERFACE

The glue-logic chip that interfaces the host to the timer/counter chip is a textbook example of PC I/O interface fit into the XC7336-7PC44 as shown in Figure 5.

It provides the logic to enable the main data-bus buffer (DQ), data-direction control (DIR), SRAM programming signals (*AM_EN and ● AM_WR), AIL-chip data-byte selection (EN-HI, EN_LO, AIL_A1, AIL_ A2), and four extra I/O read and write addresses.

I use one of the I/O addresses to control various aspects of the application, such as reset, clock enable, and single-step controls.

As I mentioned, the ISA glue-logic chip is not the epitome of EPLD design. It doesn't take advantage of the FFBs of the XC7336 nor use synchronous techniques.

It also doesn't drive the 0 WS line for faster data access-all parts of this application are fast enough for zero wait states. And, you don't have the option of 16-bit data-bus access.

There's plenty of room left for all of these advanced features to be implemented in this chip. This was a quick-and-dirty interface chip, and it worked great in 38 minutes.

It took 16 minutes to draw the symbols in the ViewLogic schematic from my scribbled notes, 5 minutes to produce the programming file, 2 minutes to program the chip, and 15 minutes to wire it in place.

TIME TO SOLDER

With all of the design in virtual space, you're ready to burn the EPLD chips and wire up a circuit.

In Part 3, I'll guide you through implementing the design in the chip and constructing a circuit for the ISA bus of an IBM-compatible computer to generate the AIL-code waveform.

I'll also discuss the steps necessary to program the SRAM waveform. ❑

David Rector is a post-doctoral fellow in the neuroscience program at UCLA. He specializes in imaging light-scattering changes in brain tissue. You may reach David at *dave@aunix.loni.ucla.edu.*

SOFTWARE

Software for this article is available from the Circuit Cellar BBS, the Circuit Cellar Web site, and on Software on Disk for this issue. See the end of "ConnecTime" for downloading and ordering information.

SOURCES

SRAM and Xilinx chips
Nu Horizons
31225 La Baya Dr., Ste. 104
Westlake Village, CA 91362
(818) 889-9911
Fax: (818) 889-1771

Marvac Dow Electronics
2001 Harbor Blvd.
Costa Mesa, CA 92627
(714) 650-2001
Fax: (714) 642-0148

DS-550, **XACT 6.0-DSVL DSVLS, FPGA, EPLD Data Book**
Xilinx Corp.
2 100 Logic Dr.
San Jose, CA, 95124
(408) 559-7778
Fax: (408) 559-7114
http://www.xilinx.com/

EPLD **programmer**
Deus Ex Machina
1377 Spencer Rd. W
Saint Paul, MN 55108
(612) 645-8088
Fax: (612) 645-0184

Deus Ex Machina programmer, Xilinx parts and software
Digi-Key Corp.
701 Brooks Ave. S
Thief River Falls, MN 56701-0677
(218) 681-6674
Fax: (218) 681-3380

IRS

413 Very Useful
414 Moderately Useful
415 Not Useful

Tuning Up

Part 3: Zerobeat Power

With this article, Ed looks after Zerobeat's final issues—detecting power failures, preserving the processor's internal RAM, and calibrating the analog parts. With this final step, we'll all be in sync and ready for communication.

FIRMWARE FURNACE

Ed Nisley

When a serious amateur radio operator invites you over, the directions always end with, "Just look for the antennas. You can't miss it."

Paul, one of the club's DX chasers, has a spiffy three-story house with a 70′ tower at the corner, topped by all manner of antennas-dipoles, beams, verticals, and what have you. His ham shack occupies the entire third story. Even *I* could tell I was at the right place!

After the usual setup confusion, he and his daughter used the prototype Zerobeat during a CW contest. He observed that, if you are spot on the sender's frequency, you're much more likely to get a response. In fact, he wants Zerobeat back for Field Day. And, he wants one of his own!

Now, if only I could talk him into becoming an ***INK*** subscriber.. .

This month, I'll explain the circuitry and firmware required to detect a power failure and preserve the 89C-1051's internal RAM. Because Zerobeat contains more analog circuitry than you've come to expect in this column, 1'11 also describe the calibration procedure.

SAVE THE BYTES!

The 20-pin Atmel 89C 105 1 at the heart of this project contains the guts of an 805 1, along with husky output ports that can drive LEDs directly. You can store programs in its 1 KB of on-chip flash memory using a handful of parts wired to your PC's parallel port.

Unlike a standard 8051, however, it has only 64 bytes of internal RAM, no external memory interface, and no

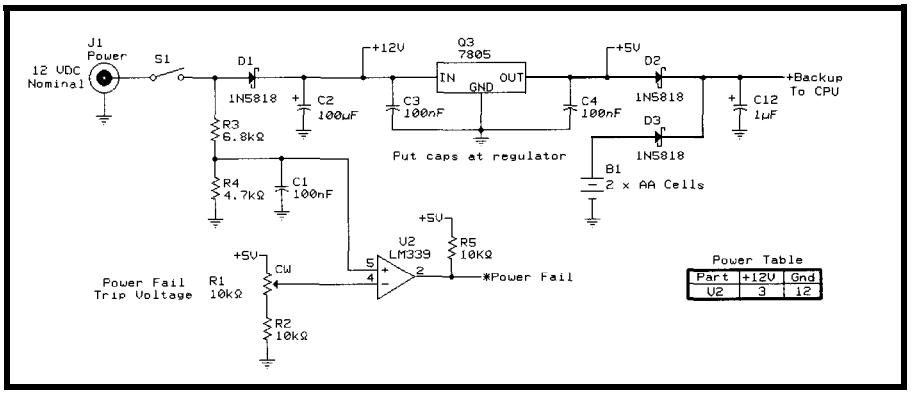


Figure I--When *the* input *power* fails, *U2b's output* triggers an *interrupt that* **gives** the *89C1051* several milliseconds warning before *the+5-V* regulator output drops. Two AA cells *maintain the* CPU's infernal RAM *until* power returns.

serial port. It may be a computer, but only for small, self-contained projects. No supercomputer applications need apply)

The Zerobeat firmware measures the incoming audio signal's period, computes the corresponding frequency, subtracts a reference frequency, and converts the result into a moving-dot LED display. The entire program uses about half of the ROM and RAM, so it fits the available resources nicely.

The 89C105 1 runs from supply voltages in the range of 2.7-6 V while drawing less than 17 mA. In power-down mode, the CPU's internal RAM remains valid even when the backup supply drops to 2 V. The chip draws only 15 µA with the clock oscillator stopped, so use whatever backup-battery chemistry you prefer.

Zerobeat must preserve only one value when the power goes off-the transceiver's audio sidetone frequency. Rather than add an external serial EEPROM or other memory device for a mere two bytes, I decided to keep the RAM alive with a backup battery.

The box I used, a slightly modified SESCOM ET-l, had enough room for a pair of 1.5-V AA alkaline cells. They should last nearly their entire shelf life in this application, and they're cheap and readily available when they do need replacing. You may substitute a small 3-v lithium cell.

Preserving internal RAM requires more than just a battery, however. After the CPU sets PCON bit 1 (i.e., the power-down bit), the crystal oscillator shuts off and freezes the hardware. Because the firmware must set the power-down bit before the supply voltage drops, it must receive advance warning of impending doom.

You need the 7805 regulator and bypass caps shown in Figure 1 anyway. All the remaining circuitry ensures the 89C 105 1 's in ternal RAM remains valid. If you're building a commercial product, it might be cheaper to throw a half-buck 93C46 serial EEPROM at the problem, skip the additional parts, and be done with it.

I built a power monitor from discrete parts simply because the rest of the circuit used only two of the four LM339 comparators. In different circumstances, a power-monitor IC might make more sense since it incorporates most of the additional parts in one DIP. Be careful you don't wind up paying more for the power monitor than you would for an EEPROM, though!

POWER PROCESSING

The 7805 regulator provides 5 VDC for the CPU and op-amp. Schottky barrier diode D1 serves two functions, protecting against an inadvertent backwards power connection and preventing C2 from discharging back into the (failing) external supply. As a result, the +5-V regulator can operate from C2 for a short time after S1 opens or the input power shuts down.

Resistors R3 and R4 form a voltage divider that reduces the nominal 12-V supply to about 5 V. U2b, part of an LM339 quad comparator, compares that to a reference voltage derived from the regulated S-V supply. You should adjust R1 so the comparator doesn't trip during normal voltage fluctuations.

My transceiver's 13.5-VDC (not, strangely, 13.8) output drops in a clean exponential curve that allows nearly 50 ms from the time U2b triggers until the 7805 output begins falling. The CPU can respond to the power-fail interrupt, turn off the LEDs, and set the power-down bit in about 10 us, so it has plenty of time to spare.

Diodes D2 and D3 isolate the CPU and backup battery from the rest of the circuits drawing power from the regulator. Don't connect anything other than the CPU to the backup battery. The forward drop across these diodes reduces the CPU supply voltage, so you should use Schottky diodes rather than standard silicon diodes.

Listing 1 shows the few lines of code required to enter power-down mode. When U2b detects a power failure, it triggers External Interrupt 0. The handler writes ones to ports P1 and P3 to reduce the current through their pull-up resistors. It then pulses bit P3.5 just before setting the power-down bit in the PCON register. You can watch that pulse on a scope to verify that the code shut down properly.

If the firmware doesn't properly enter power-down mode when the main supply voltage shuts off, the CPU continues to suck its usual 15 mA or so from the backup battery. Verify your code by measuring the supply current before and after shutdown because a teensy lithium cell can't supply 15 mA for long.

Conspicuously absent from Listing 1 is any code that calculates a checksum for the reference-frequency value. Because storing a reference occurs so infrequently, I decided to calculate the checksum once and leave it unchanged forever. That helps catch the "can't possibly happen" spurious changes that might alter a bit or two in one of the bytes.

Now that we know how to turn the CPU off, let's look at how it starts up.

START-UP EXERCISES

When normal power returns, the external Reset signal sets the CPU's Special Function Registers to their default states and starts execution at address 0000. Internal RAM remains

Listing 1--External *Interrupt 0* signals an impending *power failure. The* firmware sets Ports *1 and* 3 *high to reduce the current through the pullups, sends a confirming blip to P3.5, and turns on the power-down bit. The 89C1051 hardware shuts down before it enters the endless loop and restarts on/y after an external reset.*

```
  ORG $0003
IntOVector
  CLR A                  * power-fail input is active!
  CPL A                  * get high value for ports
  MOV P3,A               * turn off all LEDs
  MOV P1,A               * and let P1.0 and P1.1 float
  AJMP ShutDown          * handle rest of shutdown

<<< Timer 0 vector and handler fit here >>>
ShutDown
  CLR P3.5               * blip a marker output
  SETB P3.5
  MOV PCON,#POWER_DOWN * execution stops right here!
ShutLoop
  SJMP ShutLoop          * to catch CPUs without power down
```

unchanged, thus preserving whatever values were there when the CPU entered power-down mode. The 89C1051 datasheet cautions that the supply voltage must exceed 2.7 V before Reset goes active because the CPU won't run correctly below that level.

Although you're supposed to remove the backup batteries before installing the microprocessor, I plead guilty to occasionally popping it into the socket with the backup power on. The CPU can run from a 3-V supply, so without a little preventive firmware, it might draw 15 mA from the backup batteries. Ouch!

The obvious way to check for this situation requires reading the power-fail status from U2b through the I NT0 input. However, during an ordinary startup, the CPU may emerge from reset before the supply voltage satisfies U2b. A simple check might shut the CPU down immediately after the power comes on.

The code in Listing 2 dodges this problem by running through a lamp check, verifying the reference frequency stored in internal RAM, and then enabling the power-fail interrupt. If the power supply generates a reset pulse but still isn't high enough to satisfy comparator U2b, the code immediately enters the interrupt handler and shuts itself off again.

The lamp-check routine lights each of the nine tuning LEDs in sequence for about 100 ms each, which provides enough delay to stabilize the power supply. Because the LEDs and the comparator draw their power from the 7805 regulator's output, the lamp test doesn't impose any additional load on the backup battery.

Incidentally, the lamp check also verifies that you wired the LEDs in the right order. It should step from left to right, precisely backwards from the bit order.

If your LEDs light up strangely.. .well, you know what to do!

Assuming that the power supply passes muster, the LEDs indicate the results of the reference-frequency checksum test. The normal case lights the two yellow LEDs on either side of center to show a valid stored reference. If the test fails, the code uses the default 700-Hz reference and lights the two red LEDs on the ends of the display.

The first audio-frequency measurement replaces that indication with the normal moving-dot display, so you see the yellow or red LEDs blink briefly. If you start up with the transceiver audio muted, the LEDs remain lit until the first tone arrives.

DATA CHECKOUT

You must always verify data stored in supposedly nonvolatile RAM before depending on it.

The technique you use depends on how valuable you consider the data and how difficult it is to replace it. Zerobeat doesn't fall into the human-life-support category, so a simple checksum suffices for its stored frequency.

Zerobeat simply stores both the reference frequency and its one's complement in two successive 16-bit words. The start-up code verifies that the reference isn't zero (a typical value for empty RAM) and that the two values add to FFFF.

Admittedly, this method lacks some robustness, but it works well enough. A better sanity check would ensure that the frequency lies within 300-900 Hz, the typical limits for sidetone audio.

As I mentioned earlier, the code generates the checksum when it stores the frequency. That usually isn't feasible in more complex applications because the program constantly changes its critical data.

You'll probably find that your power-failure routine must verify that the most recent update is finished, then compute the checksum over a large data block. Make sure your power-failure warning occurs early enough to provide lots of time for the worst-case calculations.

In a recent 805 1 project, I wound up storing duplicate copies of a data block in battery-backed RAM. Each copy was protected by a specialized 32-bit checksum with enhanced error-detection properties. The start-up code identified a valid copy by recomputing the checksums and then sanity-checked the contents.

That firmware had a 1-ms timer interrupt handler running at all times, so I read the power-fail input during each tick. All the routines that updated the data block also disabled the timer interrupt, so the handler knew the data was in a consistent state.

It turned out that calculating the checksums occupied nearly as much time as all the other power-down processing put together.

The same techniques apply if you use a serial EEPROM or other nonvolatile memory. Make absolutely sure you can distinguish valid memory contents from the trash left by a crash during the transfer routine. Supply default values for all the parameters, then make sure you show some external, easily visible indication that all previous history has gone down the tube.

READY, SET, TUNE!

Zerobeat, being an analog project at heart, has a calibration procedure. You need only a voltmeter and milliammeter, but a scope shows the analog circuitry at work and helps track down wiring errors.

Before you install the ICs, make sure that the power supply works correctly. Connect a 9-V (not 12-V) battery to J1 and verify that the regulator produces 5 V.

Turn S1 off while you install U1 and U2, then turn R1 slowly clockwise until pin 6 of the 89C 105 1 socket goes high. That sets the power-fail trip point to 9 V, comfortably below the normal value. Remove the 9-V battery, install the 89C1051, connect J1 to your 12-V source, and fire it up.

The first time you apply power to Zerobeat, the firmware will (uh, should) discover that its internal RAM doesn't hold a valid reference frequency. It defaults to 700 Hz and lights the two red LEDs at the edges of the display until it hears the first audio tone. If the LEDs don't go on, make sure your power supply produces at least 12 VDC.

With the audio input disconnected, set R24 and R29 so that D14 and D15 are off. Plug Zerobeat into your transceiver, connect your headphones, set the CW sidetone volume to a comfortable level, and turn R25 fully clockwise to maximize the gain. Readjust R24 and R29 so those LEDs blink on when you send a string of dits and remain off when the audio is silent.

One or two tuning LEDs should go on to indicate that the firmware is comparing your sidetone to its 700-Hz reference. Press S1 to store your sidetone frequency while sending a long tone. The two yellow LEDs should blink on to show success, then the middle green LED remains on.

Tune in a strong CW signal and adjust R25 so the LEDs go on during the dits and dahs. You may also need to tweak R24 and R29. D15, the Signal LED, may flicker during weak signals, but D14, the Envelope LED, should light only during Morse-code pulses.

Your ears can probably detect weaker signals than Zerobeat's simple analog circuits. As you increase the volume to hear poor signals against the background noise, D14 and D15 may light continuously. Zerobeat can still detect and display the tones, but the tuning LEDs may not be particularly useful because Zerobeat cheerfully calculates the noise frequencies!

You can adjust R24 and R29 slightly higher under these conditions, but don't be surprised if you can't find a happy compromise between strong and weak signal work. Your narrow CW filter certainly helps reduce the noise.

The tuning LEDs go on as Zerobeat tracks the received dits and dahs. If you placed the LEDs properly, the leftmost LED is on when the transceiver is tuned below the signal and the tone is higher than your sidetone. Adjust the transceiver frequency until the middle LED goes on, then verify that the right-hand LEDs go as you continue tuning upward past the signal.

Now, with the power still on, install a pair of AA cells in the backup

Listing 2—*The start-up code sets up the hardware, blinks the LEDs, and verifies the reference frequency in infernal RAM before enabling External Interrupt 0. If the power supply is high enough to start the CPU but below the level set by R1, the Int 0 handler enters power-down mode when this code sets EXO.*

```
* set up the hardware
  MOV   SP,#StackBase-1       * put stack after all variables
  MOV   TCON,#$04             * Int 1 edge-triggered
  MOV   TMOD,#$01             * Timer 0 in 16-bit mode
  MOV   IE,#$80               * allow interrupts when ready

* do a simple lamp check
  MOV   R1,#$01
  MOV   R2,#$00
  MOV   R3,#9

LampCheck
  MOV   A,R1
  MOV   B,R2
  ACALL SetLEDs
  MOV   A,R1
  MOV   B,R2
  MOV   R0,#1
  ACALL ShiftLeft
  MOV   R1,A
  MOV   R2,B
  MOV   B,#0
  MOV   A,#100
  ACALL Delay1ms
  DJNZ  R3,LampCheck

* check for valid stored reference frequency
  MOV   A,#REF_OK             * assume it will be valid
  MOV   B,#=REF_OK
  ACALL SetLEDs
  MOV   A,RefFreq             * make sure it is not zero
  MOV   B,RefFreq+1
  ORL   A,B
  JZ    InitRef
  MOV   A,RefFreq             * do the copies add to FFFF?
  ADD   A,RefInverse
  CPL   A                     * only FF becomes 00
  JNZ   InitRef               * low byte is bad
  XCH   A,B
  ADD   A,RefInverse+1
  CPL   A                     * only FF becomes 00
  JZ    RefReady              * high byte is OK

InitRef
  MOV   RefFreq,#REF_DEFAULT * set default reference
  MOV   RefFreq+1,#=REF_DEFAULT
  MOV   RefInverse,#~REF_DEFAULT
  MOV   RefInverse+1,=~REF_DEFAULT
  MOV   A,#REF_NG             * show that we used the default
  MOV   B,#=REF_NG
  ACALL SetLEDs
RefReady

* enable Int0 interupt to catch power failures
* if Int0 is still low, IE0 will be high and power is no good.
* Shut down immediately
  SETB  EX0
```

battery holder with a milliammeter in series. I tucked a small piece of double-sided circuit board between one cell and its contact. Then, I clipped the test leads to wires soldered to the foil on each side.

The meter should read 0 µA when S1 is closed and less than 20 µA when S1 is open. If the batteries supply about 15 mA with S1 open, the power-fail detection circuitry did not give the controller enough warning to enter power-down mode. Adjust R1 to suit.

Each time you turn Zerobeat off and on again, the firmware should find a valid frequency in internal RAM and light the two yellow LEDs. If your transceiver has a DC power outlet or if you wire Zerobeat to the same supply as the transceiver, it needs no further attention except while you tune a signal.

RELEASE NOTES

It turns out that some transceivers, mine included, have a slight offset between their sidetone frequency and the tuning oscillator. I decided that a 20-Hz difference wasn't enough to justify opening the transceiver and tweaking its BFO. But, if you find that Zerobeat consistently indicates correct tuning slightly off from a sensible setting, that may be the cause.

When you can hear both sides of an exchange between two other stations, Zerobeat shows you the frequency difference between the two transmitters. Most hams do indeed have better ears than I, but I've monitored some conversations that were off-scale in both directions! The tuning LEDs span about 280 Hz-enough that even I can hear the difference.

I uploaded the Zerobeat source and hex files for the column in *INK* 73. That column also lists sources for some of the parts you need for this project.

THE CLOSING BRACE

Next month brings something altogether different: no Firmware Furnace! There are times in life when you're pushed to make decisions and take new directions.

Thanks to all of you who suggested topics, pointed out the error(s) of my ways, and generally kept me on my toes. It's particularly gratifying to hear from folks who used my columns as the basis for their own designs. I've learned a lot while researching and building these projects, so the benefit works both ways.

While I'm not sure what comes next, I plan to publish some books based on some of my favorite columns. So, we'll surely meet again further along the bitstream. Enjoy! ❑

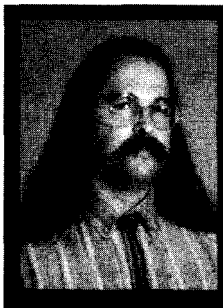
***Ed Nisley** (KE4ZNU), **as Nisley Micro Engineering, makes small computers do amazing things. He's also a member of Circuit Cellar INK's engineering staff. You may reach him at** ed.nisley@circellar.com **or** enisley@ibm.net.*

IRS

416 Very Useful
417 Moderately Useful
418 Not Useful

Just One Mile More

Creating a NC-based Pedometer



Exercise is a must for desk jockeys, so Jeff is an avid jogger. This month, he designs a pedometer that uses Micromint's PicStic to keep him informed of his progress.

FROM THE BENCH

Jeff Bachiochi

When you hear about the Olympic torch, what comes to mind? For me, it conjures up a view of a runner carrying a flaming torch high overhead and looking like Lady Liberty.

Originating from the site of the first Olympic Games in Greece, torch bearers share the honor of carrying the flame to the site of the next Games. This flame is transferred to a giant torch marking the opening of the latest round of worldwide competition. This summer, the Games were hosted by the U.S. in Atlanta, Georgia.

The torch relay came through our part of Connecticut on Father's Day and was played up by the media for all it was worth. For about $275, you too could own an Olympic torch (knock-off). There were a bunch of takers.

These enthusiasts passed their piece of history to all the youngsters gathered to watch the torch whisk by. I saw a special pride in their eyes as just for a moment they held the prize aloft.

The Olympic torch was not only carried by runners but by cyclists and wheelchair athletes. I was inspired.

ENOUGH EXERCISE?

For me, running is a satisfying way to exercise. My doctor says exercise is a good way to keep cholesterol in check, which is good for someone who likes burgers and pizza.

"You should exercise every day. Elevating your heart rate keeps your heart strong and healthy," he says. "And one more thing. Lose 15 pounds or gain a few inches in height."

When I run, I lose track of time and I find myself thinking about every thing but running. Mind you, this may be the body's way of distracting itself while you inflict unnecessary pain.

Other than a watch, there are few ways to monitor your progress-unless you choose to run on a track. It's helpful to track your time, present overall speed, and distance traveled. While treadmills can do this for you, I find them boring. I miss the fresh air.

This month, I put together a little pedometer which mounts on the laces of my running shoe and gives me constant feedback of speed and distance via a two-digit hexadecimal display (see Photo I).

The display updates on each step and rotates through four catagories: AS (average speed), PS (present speed), dt (distance), and Et (elapsed time).

Such devices must be small and lightweight. They shouldn't add any significant mass to the sneaker. To accomplish this, I chose a PicStic as the core.

PICSTIC MICROCOMPUTER

Micromint's PicStic is based on Microchip's PIC 16C84 processor. In addition to the processor, it includes a surface-mount voltage regulator, reso-

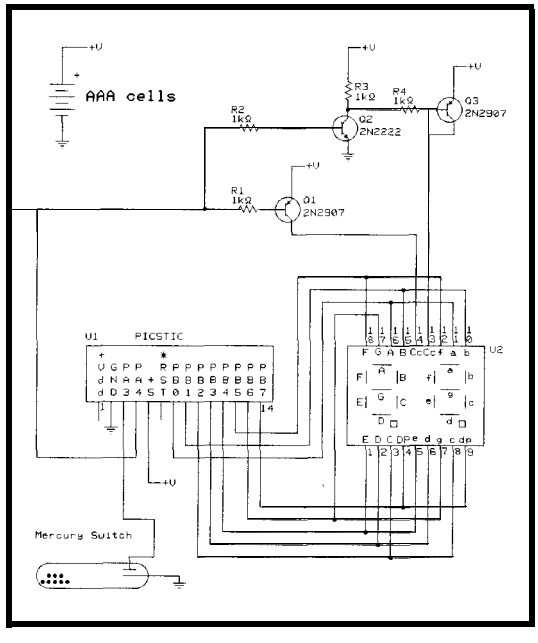


Figure 1—*A PicStic processor alternately drives the twin digits of a hexadecimal display.*

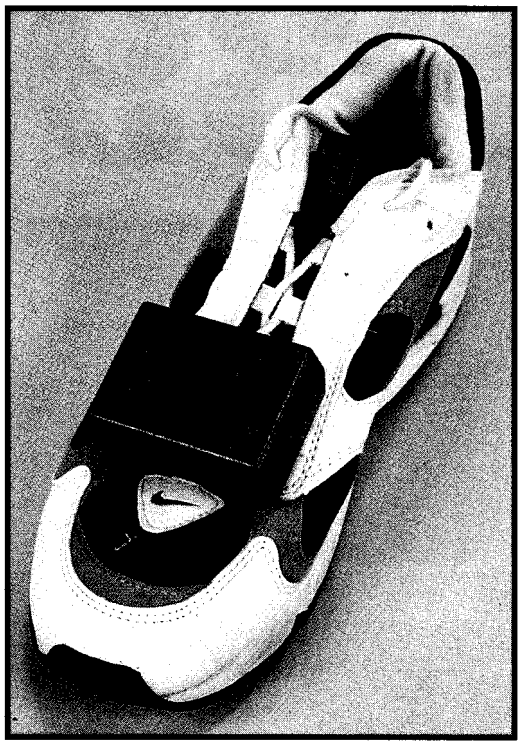


Photo I-Powered *by* four AAA bafferies, the *PicStic* drives *the digital* display which gives me *feedback about how hard I should be sweating.*

nator, and optional 12-bit ADC or RTC to create a microcomputer on a narrow 14-pin SIP (see Figure 1).

PicStic accepts assembler files from any PIC assembler. I use microEngineering's PBASIC (which compiles to assembler]. That way, I can write BASIC code and avoid assembler.

It's not that PIC assembler is difficult, but for me, the fun is *not* in the programming but in the hardware. So, why not make the software development as painless as possible?

PicStic can directly drive LEDs with the digital I/O port, so I wired each seven-segment display (seven segments and a decimal point) to the eight-bit I/O of Port B. A single output bit on Port A is the control line for the display's common anode connections. The output bit drives some transistors which supply the master current to each of the separate display digits.

The NPN transistor's output doesn't drive the display but inverts the control signal for the second PNP transistor which supplies current for the right digit. This allows the two digits to be powered alternately (only one digit is active at a time). By alternating the digit data and control bit fast enough, both seem to be illuminated at the same time.

I chose the PicStic with the RTC option so I could track the elapsed time. Each time the processor resets, the clock is reset to "OO:OO:OO".

From this point on, all computations are made from a "00" hour, "00" minute, and "00" second starting point. The elapsed time is displayed (during states 9, 10, and 11) as "Et" "00." (minutes) "00" (seconds).

GOING THE DISTANCE

At this point in the project's development, the runner's pace is hard coded as a constant in inches. I might add provisions for setting this through a push button later. All computations are based on this figure (given in inches), so you should make it as accurate as possible.

I used the high school track to determine my pace length. Uneven territory messes up an otherwise constant stride. Although my stride decreases while running up a hill, it increases going downhill. So, by starting and ending at the same location, my short paces cancel my long ones.

A single bit identifies when the module receives a jolt from a sneaker hitting the ground. While the PicStic waits for this to happen, the display alternates between the left and right digit's data for whatever state it's in.

Once a step is identified, a pace is added to the inch counter. This counter is checked to see if 1/100 of a mile (i.e., 634") has been reached.

If so, the distance is incremented by 1 (i.e., 1/100 of a mile). This distance displays during states 6, 7, and 8 as "dt00.00" (up to 99.99 miles).

FULL SPEED AHEAD

Once distance is established, *Et* is used to calculate the average speed. The speed displays during states 0, 1, and 2 as "ASOO. 00" (up to 99.99 MPH-you never know when I'll hit Superman-dom).

The *Et* which is read from the RTC as month, day, year, day of week, hours, minutes, and seconds is converted to total seconds using:

total seconds = (hours x 3600) + (minutes x 60) + seconds

The distance traveled (in 1/100s of a mile) is divided by the total elapsed time (in seconds). This speed in hundredths of a mile per second is multiplied by 3600 to get miles per hour.

Note that PBASIC does *not* do floating-point math, even though it handles a 16- x 16-bit multiply and passes the remainder fraction in a division. So, be careful to avoid large overflows and underflows in equations.

Present speed is calculated much the same way. It is the average speed during the last 12 paces (one full cycle of the state machine) as opposed to the total time running.

The present speed displays during states 3, 4, and 5 as "PS00.00" (up to 99.99 MPH). It is calculated using the time in seconds since the last pass through state 6. The distance is therefore fixed at I2 paces.

state0	Left digit is set as "A"
	Right digit is set as "S"
	Return
state1	Get_Clock
	Calculate MPH (total distance/total time)
	Left digit is set as 10s of MPH
	Right digit is set as 1s of MPH
	Return
state2	Left digit is set as 10ths of MPH
	Right digit is set as 100ths of MPH
	Return
state3	Left digit is set as "P"
	Right digit is set as "S"
	Return
state4	Get_Clock
	Calculate Time since last State4
	Calculate MPH (12 paces/time)
	Left digit is set as 10s of MPH
	Right digit is set as 1s of MPH
	Return
state5	Left digit is set as 10ths of MPH
	Right digit is set as 100ths of MPH
	Return
State6	Left digit is set as "d"
	Right digit is set as "t"
	Return
state7	Left digit is set as 10s of total distance
	Right digit is set as 1s of total distance
	Return
State8	Left digit is set as 10ths of total distance
	Right digit is set as 100ths of total distance
	Return
state9	Left digit is set as "E"
	Right digit is set as "t"
	Return
State10	Get-Clock
	Left digit is set as 10s of elapsed minutes
	Right digit is set as 1s of elapsed minutes
	Return
State11	Left digit is set as 1 Oths of elapsed seconds
	Right digit is set as 100ths of elapsed seconds
	Return

Table 1 --The twelve-step *PBASIC program displays the data on the* twin hexadecimal digits of *the* pedometer.

This total distance is divided by 12 to get the distance in feet. This value is divided by 52.8 (5280 feet per mile) to get the portion of a mile, and then multiplied by 3600 for miles per hour.

For an approximate calculation, you can multiply the distance of 12 paces by 6-which is the approximate value of 3600 divided by 12 times 52.8.

Regardless of which way you find speed, divide it by the actual *Et* (time for the I2 paces) for present speed.

TRIGGERING THE PACE

Many different kinds of sensors could be used to jog the pace. Pressure sensors could be built into the running shoe similar to the Pump. (The Pump is basketball footwear which acts like a built-in blood-pressure cup expanding to support your foot.)

The bladder could have a sensor which monitors pressure changes. A microphone or piezoelectric disc could produce an output each time it heard or felt your shoe hit the pavement.

Of course, the simplest solution might be to place a mechanical switch in the sole of your shoe. However, this does have portability problems.

To keep it simple yet portable, I added a mercury switch which floods the contacts when I thrust my foot forward.

The seven-segment display has a decimal point on the left side of the digit, so it reads "00." and "00" rather than "00" and ".OO''-a bit less clear but acceptable. The decimal point is turned on by adding 80h to the lookup table value when printing the right hand digit in states 1, 4, 7, and 10.

Output pin PA4 toggles continuously while idle between states. This pin drives the master current switch for the left digit and, once inverted, drives the master current switch for the right. Thus, each digit is turned on and off from the same pin. Digit data is synchronized for each cycle of the control pin.

I turn off the display prior to toggling the control line to prevent a faint shadow of the opposite digit from being visible while the data is updated.

A couple of microseconds gives some illumination, and blanking the data eliminates the shadow. When any of the eight data bits on Port B are logic zero while the master current transistor for a digit is on, the connected LED segment illuminates.

The lookup table consists of the values which-when written to Port B-produce"O-9" and "A", "P", "d", "E", "t", and a blank. The decimal point is the most significant bit.

FINISH LINE

Most of the time, the processor checks for a step loop and multiplexes the display between the two digits.

PBASIC on the PicStic is fast. A single-digit update loop (which includes checking for a step) takes about 200 µs. Eventually, a step is detected.

After updating the distance, the present state increments, and the program continues through one of the twelve states. Table 1 shows an overview of what happens in each state.

Although I used practically all the variable storage in the PicStic for the project (the RTC routine alone requires 7 of the 22 available bytes of variable storage), the compiled program requires only about three quarters of the available 1 KB of space.

It's easy to see that the PicStic's size-price-performance ratios are something we can all smile at. Embedding micros has just gotten a whole lot friendlier. PicStic wins the gold.

Jeff Bachiochi (pronounced "BAH-key-AH-key") is an electrical engineer on Circuit Cellar INK's engineering staff. His background includes product design and manufacturing. He may be reached at jeff.bachiochi@circellar.com

SOURCES

PicStic, development kit, PBASIC compiler
Micromint, Inc.
4 Park St.
Vernon, CT 06066
(860) 871-6170
Fax: (860) 872-2204
http://www.micromint.com/

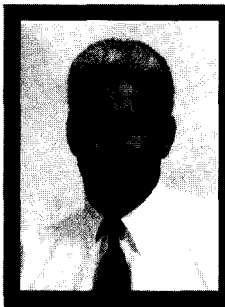
IRS

419 Very Useful
420 Moderately Useful
421 Not Useful

SILICON UPDATE

Tom Cantrell

Stop Me Before I Crash Again



Tom doesn't trust built-in watchdogs. If the code for the chip goes down, what stops the code for the timer and interrupt from being bad too? TelCom that's who. Come check out their supervisory controllers.

Though taken for granted, it's actually quite amazing that PCs and embedded micros work at all. Millions of transistors have to do the right thing millions of times per second day in and day out.

Come to think of it, besides the well-known Pentium floating-point foopah, the last time anyone had to worry about hardware was in the early days of DRAMs (i.e., more than 10 years ago).

I can remember designers struggling with finicky timing, signal reflection foibles, and even alpha particles. Many simply punted, designing in error detection or correction circuits, a close-to-home example being memory parity on the PC.

Now, I obviously don't follow the desktop world real closely given the vintage of all my machines. However, I did discover during a recent PC RAM upgrade that parity protection apparently disappeared somewhere way back when-at least in the case of my no-name clone.

I guess I wasn't so disturbed by the unexpected lack of parity protection as by the fact that I-and presumably a lot of others-dutifully bought x9 SIMMs. Are there really zillions of PCs carrying around terabits of excess baggage?

Perhaps the improving reliability of DRAM combined with the intense cost pressures in the PC biz justifies the elimination of parity protection. However, I suspect the major reason no one will miss it is that the PC software didn't deal with parity errors gracefully anyway. Why bother detecting a bad bit if the software's response-basically, hang the machine—makes the situation worse!

Yes, it's easy to dismiss hardware reliability issues given the fact that software is so flaky. Whether it's spacecraft lost because a ";" should have been a "," or an x-ray machine that nuked unfortunates due to a WYSI-Not-WYG interface bug, to the ever-more-imminent year-2000 rollover fiasco, it seems software is usually to blame for the more flagrant foul-ups.

Since prospects for more reliable software don't seem promising, hardware folks need to do more, not less.

This is especially true on the embedded front where the stakes are

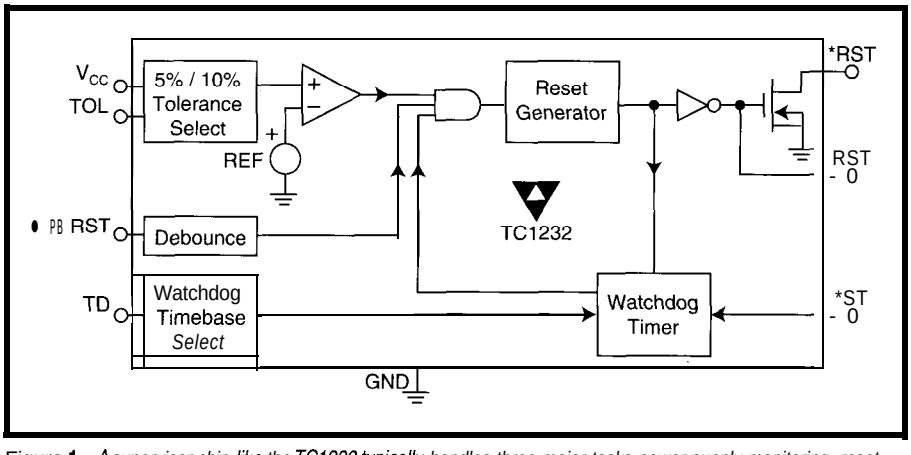


Figure 1—*A supervisor chip like the TC1232 typically handles three major tasks-power-supply monitoring, reset generation, and watchdog timing.*

somewhat higher than on the desktop. If your car's antilock braking system is as flaky as your PC, your next crash could be your last.

POWER ME UP, SCOTTY

So-called supervisor ICs have evolved over the years to provide a measure of robustness for even the simplest micro applications. As silicon marches on, the number of chips and suppliers is growing.

TelCom Semiconductor, a recent spin-out of Teledyne, offers a line of parts that illustrates the variety of features available.

The most basic function of a supervisor IC is power-supply monitoring integrated with reset-signal generation. The best way to see why these features are useful is to consider a design without them.

The question is, exactly what happens as the power-whatever the source-goes on and off?

The answer is: absent supervision, anything can happen. Unfortunately, "anything" can mean a number of tragic outcomes.

In presupervisor days, most designers just hung an RC on the reset pin and hoped for the best. However, those who've done it that way know the approach isn't bulletproof.

Often, such a system won't start at powerup. Who hasn't performed the "unplug it, wait a second, try again" ritual? A setup that works well with a particular power supply might not work at all with a different one.

Of course, the scary part is that "fail to start" is a naively optimistic description of what's going on. Fact is, no IC supplier can or will guarantee behavior in the twilight zone.

Most embedded applications either don't have the luxury of a reset switch or one wouldn't be a good idea in any case.

They also might not have a way to tell the user-or even to determine—whether anything is amiss. Indeed, one of the scariest failure scenarios has the system appear to work, though a few scrambled bits are lurking, waiting to pounce.

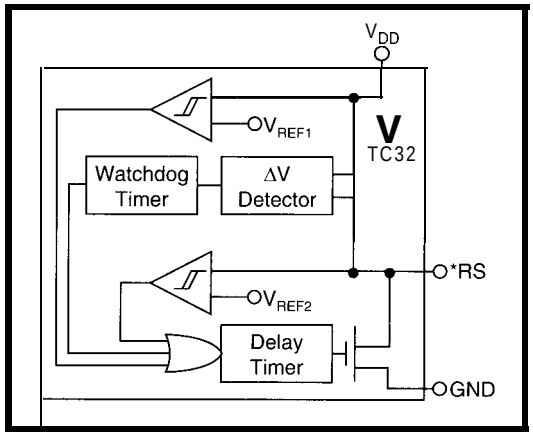


Figure 2—*Most* of the *TC1232's* functionality fits *into the TC32's* smaller *and lower cost package.*

Traditionally, powerdown has been a less critical issue. However, the increasing tendency to use nonvolatile data memory to preserve information across power cycles has an impact. This is most notably true for battery-backed SRAM, which is only one spurious write cycle away from trouble.

Besides the power-on and -off boundary conditions, the specter of brownouts during operation can't be ignored. Here, the RC hack is going to drop the ball since the gap between valid operation (typically 4.5 V minimum) and a valid reset (e.g., 0.8 V maximum for active low) leaves plenty of room for mischief.

Once again, a chip may appear to ride out a brownout, although it's corrupted. Multichip systems offer the intriguing possibility that different chips will be in different states of disrepair.

WHO WATCHES THE WATCHERS?

Guaranteeing the processor is either getting the power it needs or being held in reset-and nothing in-between-ensures that the system gets started down the right path.

Nevertheless, any number of hardware and software hazards line the route. A watchdog timer acts as kind of a backseat driver to keep the micro on the straight and narrow.

Not to besmirch the trustworthiness of anyone's particular design, the concept of a watchdog is basically to let the system out on parole and require it to check in periodically. If the micro doesn't show up on time, the alarm (i.e., reset) goes out.

Any micro with an onboard timer that generates an interrupt is able to implement a similar function. The micro must check in by reloading the timer before the timer elapses. If it doesn't, the occurrence of the timer interrupt signals something is amiss.

The interrupt handler can try to get things back on track or, with the aid of an output pin and some glue logic, even make the CPU reset itself. Most recent vintage micros have at least this level of timing capability, and many even include dedicated watchdogs.

However, I've always had a queasy "fox guarding the henhouse" feeling about onboard watchdogs. It seems paradoxical to trust the chip to catch itself being untrustworthy.

If software can have bugs, doesn't that include the watchdog code as well? If bits can get scrambled, what makes the ones that turn off the timer or mask the interrupt immune?

It seems somewhat safer to use a separate chip like the TC1232 (see Figure 1). Offered in an eight-pin DIP or SOIC package, the chip integrates the traditional supervisory triad of power monitor, reset generator, and watchdog timer. At only $1.98 in 1000-piece quantities, it seems a small price to pay.

Starting with the outputs and working back, the TC1232 features separate high and low RST outputs. The active-

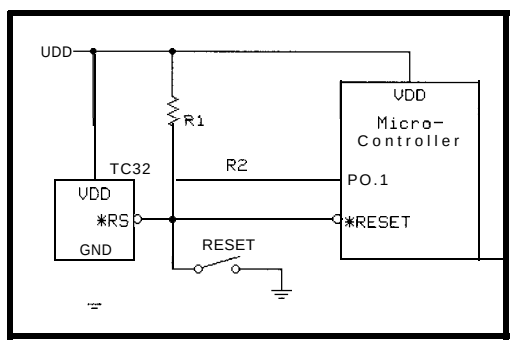
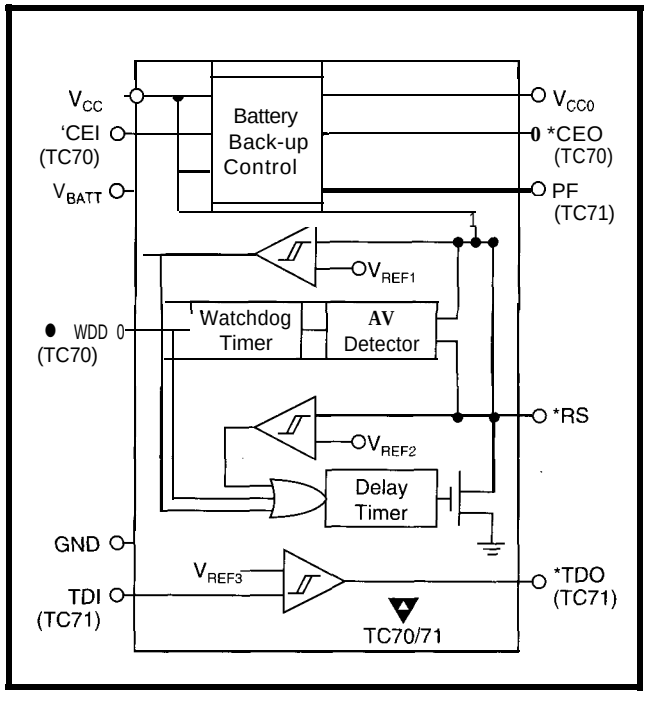


Figure 3—*The TC32* ● RS *(RESET/STROBE) input serves triple duty as an output, push-button input, and watchdog-strobe input*

Figure *4-The latest supervisor ICs, such as the TC70 and TC71, add functions to accommodate SRAM including battery switchover, write protection, and fheshold detection.*



low one is open collector, allowing external wire-ORing with another reset source. It features an internal pullup as well, eliminating the need to add a resistor.

There's also a separate push-button (*PB RST) input with built-in debouncing. Whatever the circumstances, the reset generator always delivers a healthy 250-ms pulse ensuring no false starts.

On the input side, V_{CC} is monitored to within 5% (4.75 V) or 10% (4.5 V) tolerance selected by the level on the TOL pin. The other two inputs are associated with the watchdog timer.

The TD pin selects the watchdog interval, cleverly encoding three choices (150 ms, 600 ms, or 1.2 s) on a single pin (i.e., it's grounded, connected to V_{CC}, or left open). Within the selected interval, the micro must strobe the *ST input or the TC1232 initiates a reset.

LITTLE BROTHER IS WATCHING

Building on the foundation of first-generation chips like the TC1232, suppliers are moving in two directions as silicon marches on. One is to offer most of the traditional functionality for less, as does the aptly named TC32 Economonitor.

As shown in Figure 2, the TC32 manages to offer nearly all the functionality of the TC 1232 but crams it into a tiny [TO-92 or SOT-223) package. This, coupled with a price cut ($0.86 in 1 OOO-piece quantities), make supervision attractive for even the lowest cost applications.

Now, 3-pin supervisors aren't a brand-new concept with most of the usual suspects (e.g., Maxim, Dallas, etc.) offering them.

But, the TC32 is the only one I know of that's able to get all three basic functions (i.e., supply monitor, reset, and watchdog) into the small package.

Of course, stuffing the entire functionality of the 8-pin '1232 into the 3-pin '32 isn't possible. Since two pins have to go for power and ground—nobody's figured out a way around that yet-it would require 5 pins' worth of '1232 to converge on an overworked

- RS pin of a '32.

V_{DD} (V)	R1 Std. Value (k)	R2 Std. Value (k)	Nominal V_{STL} (V)	Min. V_{STL} (V)	Max. V_{STL} (V)
5.0	7.5	6.2	2.25	2.02	2.89

Table *I-The resisfor network must establish the proper watchdog strobe voltage considering the tolerances of fhe components involved (R1, R2 tolerance = ±5%; V_{DD} tolerance = ±5%).*

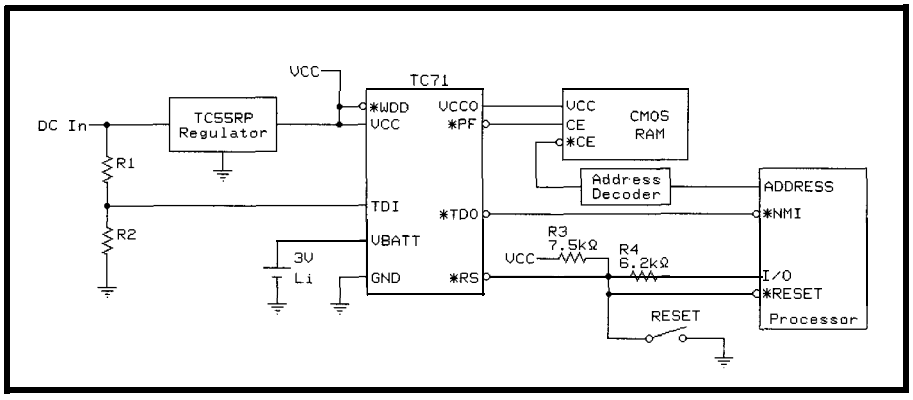


Figure 5—*By adding a threshold defector, the TC71 monitors the input to (and output from) the regulator, giving advance warning of power cutoff.*

So, the first thing that goes is the TC1232's active-high RST output. I suspect it won't be missed since most of the micros I've dealt with have active-low reset.

The next sacrifice is the TC 1232's TOL (5% vs. 10%) pin so the TC32 trip level is fixed at 4.5 V (i.e., 10%). Along the same lines, the watchdog period is hardwired, dispensing with the TC1232's TD pin.

Whether the watchdog period is fixed or programmable, take note of the detailed specs since the variability dictates both the check-in requirements for your micro and how long the leash is.

For instance, the spec for the TC32 watchdog period varies from 500 ms minimum to 900 ms maximum, with 700 ms being typical. Thus, the designer must guarantee the micro shows up every 500 ms while being forewarned that it could run wild for up to 900 ms.

Surprisingly, the downsizing stops here with the TC32 single *RS pin [aided by a couple of resistors) able to handle reset output generation, push-button connection, and the watchdog-strobe input as shown in Figure 3.

As before, the TC32 drives reset low at powerup and holds it low during power glitches. At the same time, it accepts and debounces a switch (or other signal) input and produces a clean reset.

Strobing the watchdog relies on the trick of driving the *RS pin to an intermediate level, established by the driver characteristics and resistors, that's low enough to be recognized as a strobe but not low enough to trigger a reset. Specifically, it's somewhere between 2 and 3.5 V.

Table 1 shows an example resistor configuration assuming 5 % voltage and resistor tolerances and a TTL-like V_{OL} = 0.4-V maximum driver.

Besides assuring the nominal strobe voltage (i.e., 2.25 V) is within the allowed range, you have to account for the worst-case extremes.

The highest output occurs with power and R2 at +5%, R1 at –5%, and V_{OL} = 0.4 V, and the minimum with power and R2 at –5%, R2 at +5%, and V_{OL} = 0 v.

WE WANT YOU, BIG BROTHER

While the TC32 delivers most of the TC1232 bang for fewer bucks and less board space, the TC70 and TC71 [at $1.86 in 1000-piece quantities) go the other direction by reverting to the 8-pin package but packing a lot more stuff in.

Figure 4 shows the chips' internals and identifies the pin differences (4, 5, and 6) that distinguish the '70 and '71. Let's start by covering the pins that are the same.

Along with power and ground, the 'RS pin works the same on the '70 and '71 as it does on the '32 (including the resistor-voltage-divider trick) to combine reset output and push-button and watchdog-strobe inputs.

A major new addition is automatic power switchover to accommodate the needs of battery-backed logic, most notably CMOS SRAMs.

The way it works is that whichever input voltage (i.e., V_{CC} or V_{BATT}) is higher gets switched to the output (i.e., VCCO).

This guarantees (assuming the battery is up to snuff) that the RAM never experiences an amnesia-inducing power glitch. The switchover circuit incorporates 20mV of hysteresis to prevent chattering at the margin.

From this point on, the '70 and '71 diverge slightly (i.e., pins 4, 5, and 6). However, both deal with one more aspect of SRAM backup, they just do it in slightly different ways.

You'll remember how the power-monitor function holds the processor in reset, preventing spurious SRAM writes and the RAM is backed by VCCO as V_{CC} fades. However, there's still the potential for trouble if the SRAM control lines (e.g., *CE and *WE) aren't held in check.

The '70 offers a no-glue solution in the form of *CEI (*CE In) and *CEO (*CE Out). As you might guess, you simply insert this pair into the path from your micro to the SRAM *CE pin.

Under normal operation (i.e., V_{CC} is greater than 4.5 V), 'CEO simply tracks *CEI, but watch out for the 30-ns maximum prop delay if your timing is tight. However, when the power is cut, *CEI is ignored and *CEO driven to VCCO to batten down the hatches.

The '71 accomplishes more or less the same task with a single ● PF (Power Fail) output pin. Many SRAMs have an extra CE pin that makes a good home for ● PF, or it can be used to gate external decode logic.

This leaves the **'71** with two pins (TDI and *TDO) that provide a useful threshold detection feature. Very simply, the *TDO (Threshold Detector Out] pin goes low whenever the voltage on TDI is less than 1.3 V.

As shown in a typical TC71 -based design (see Figure 5), TDI (via a suitable voltage divider) can monitor the input side of a regulator, giving early warning on *TDO long before V_{CC} goes out of spec.

BEWARE OF THE DOG

What about the last pin on the '70? • WDD (Watch Dog Disable) is intended to minimize distraction during prototyping and debug. And, it does exactly as it should.

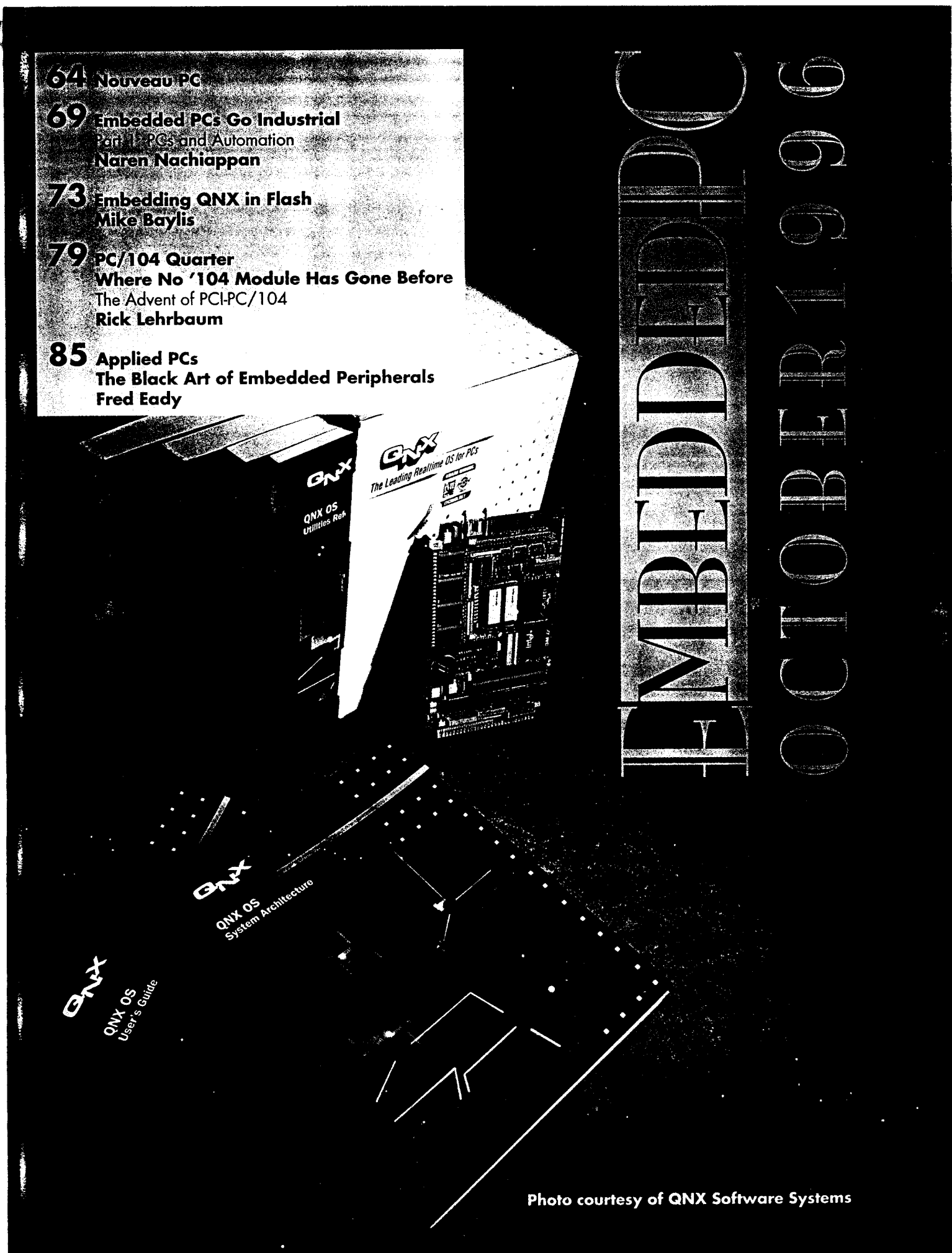
There might be other legitimate circumstances for disabling the watchdog, but be real careful not to let it bite your hand. ❑

Tom Cantrell has been working on chip, board, and systems design and marketing in Silicon Valley for more than ten years. He may be reached by E-mail at tom.cantrell@circellar.com, by telephone at (510) 657-0264, or by fax at (510) 657-5441.

SOURCES

TC1232, TC32, TC70, TC71
TelCom Semiconductor, Inc.
1300 Terra Bella Ave.
Mountain View, CA 94043-1800
(415) 968-9241
Fax: (415) 967-1590

422 Very Useful
423 Moderately Useful
424 Not Useful

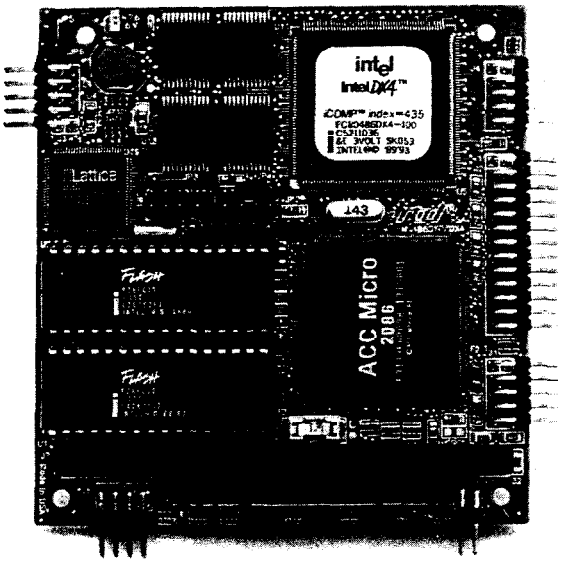


EMBEDDED PC

OCTOBER 1996

Photo courtesy of QNX Software Systems

80486 CPU MODULE



The **CMV486DX4 cpuModule** from Real Time Devices delivers the performance of high-end PC/AT '486 desktop computers and the functionality of embedded DOS controllers on the PC/I 04 form factor for industrial control. It can be stacked with RTD's PC/I 04 *utilityModules, dataModules,* and expansion cards to create complete systems for embedded control and data processing.

These cpuModules feature a high-performance 100-MHz Intel 80486DX4 processor, internal math coprocessor, 32-bit memory bus, and PC/104 bus. They have 8-MB DRAM onboard and two 32-pin solid-state disk sockets for **2-MB EPROM or** 1-MB flash, SRAM, or NVRAM. They also include two RS-232 serial ports, bidirectional parallel port, AT keyboard and speaker port, and a watchdog timer. The unit operates under **MS-DOS, ROM-DOS, DR-DOS,** Windows, OS/2, UNIX, QNX, RTXC, and other real-time operating systems.

The services and functions provided by the embedded BIOS ensure PC/AT compatibility. BIOS enhancements provide programmable Quick Boot, virtual device support, and solid-state disk support for EPROM, flash, SRAM, and NVRAM including flash-file system for 5-V flash. Virtual device support lets you use the keyboard, video, and floppy and hard disks on another PC-compatible computer through the serial port. A nonvolatile 1024-bit configuration EEPROM stores the system setup and provides 5 12 bits to the user.

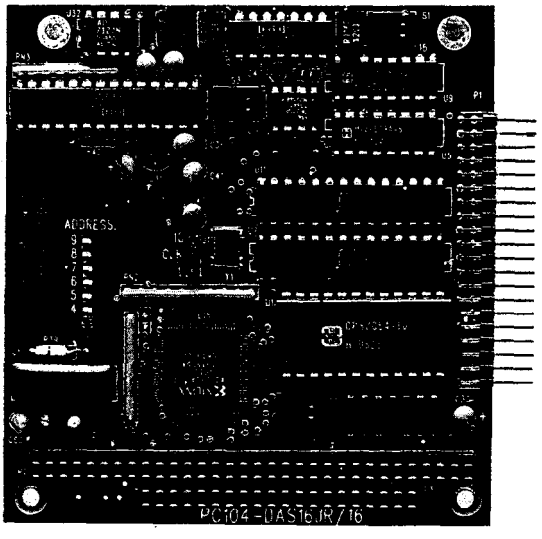
The CMV486DX4 sells for **$810** in 100-piece quantities.

Real Time Devices, Inc.
200 Innovation Blvd. • State College, PA 16804 • (8 14) 234-8087 • Fax: (8 14) 234-52 18

#510

16-CHANNEL A/D CARD

The **PC1 04-DAS16/JR**, available in 12- or 16-bit versions, is a perfect solution for analog data acquisition using the compact PC/I 04 platform. Software-selectable gains allow for input-signal ranges from **± 10** V to 20.625 V bipolar and from O-I 0 V through O-O.625 V unipolar. Triggering to **begin the transfer and acquisition cycle can be performed through software, internally paced, or externally triggered.**



The 150-kHz PC 104-DAS16/JR employs a 5 12-sample FIFO buffer to ensure the transfer of large blocks of data with no missing samples. Sampling of A/D signals is switch selectable for 16 single-ended or 8 differential channels. Three 16-bit counters are available for event counting, frequency measurement, or A/D pacing. Eight lines of digital I/O are configured as four inputs and four outputs **for sensing a contact closure or controlling a discrete device.**

The 12-bit **version** sells for $399 and the 16-bit version for **$499.**

Computer Boards, Inc.
125 High St., Ste. 6
Mansfield, MA 02048
(508) 261-I 123
Fax: (508) 261-1094

#511

Nouveau PC

edited by Harv Weiner

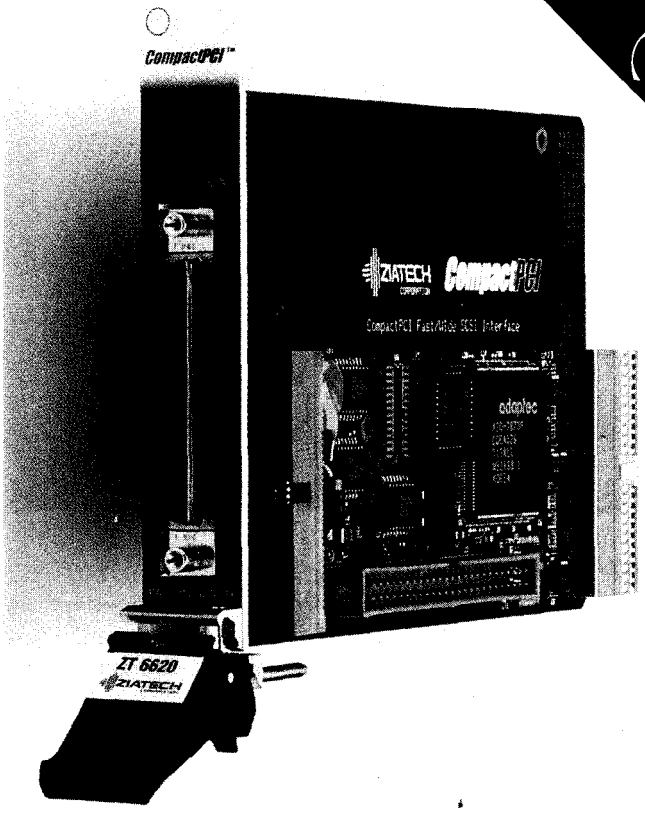
FAST AND WIDE SCSI INTERFACE

A Fast and Wide SCSI interface for the new CompactPCI industrialcomputer standard has been introduced by Ziatech. Based on Adaptec RISC technology and a 3U form factor, the 216620 Wide SCSI Interface maximizes system I/O performance by merging the Wide SCSI protocol with the fast CompactPCI bus. It offers SCSI data transfer rates up to 20 MBps and CompactPCI burst-transfer rates up to 132 MBps. Fast and Wide SCSI provides connections to external SCSI devices such as RAID disks, CD-ROMs, and other high-speed peripherals.

The ZT6620 is compatible with all major operating systems and SCSI-2 and -3 peripherals, as well as industry-standard application software. The design is also software-compatible with the Adaptec AIC-2940W Wide SCSI interface.

The ZT6620 features a BIOS-resident select utility which eliminates the need to manipulate jumpers and terminators for all but the most specialized applications. The interface features two 16-bit-wide SCSI-3 connectors (one internal and one external) and one 8-bit SCSI-2 connector. Up to 15 devices can be supported on one of the two 16-bit connectors and up to 7 devices can be supported on the 8-bit connector, depending on DOS and BIOS limitations.

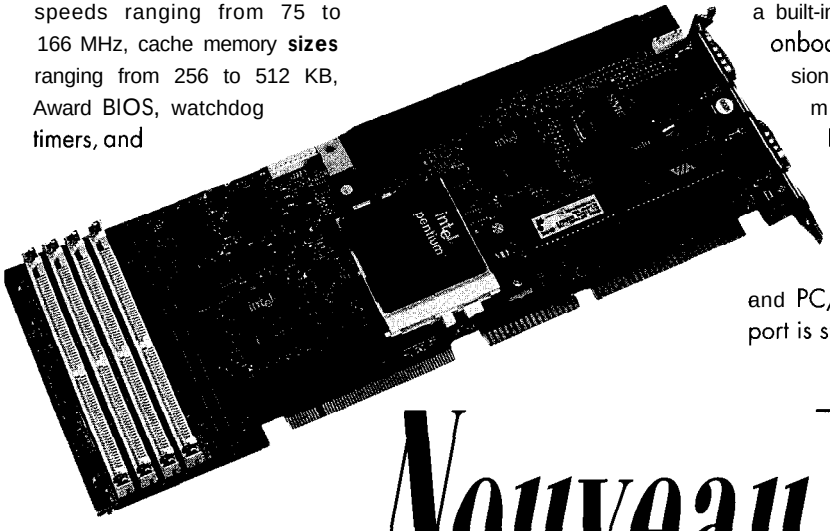
The ZT6620 Wide SCSI Interface sells for $495.



Ziatech Corp.
1050 Southwood Dr. • San Luis Obispo, CA 9340 1
(805) 541-0488 • Fax: (805) 541-5088
http://www.ziatech.com/ **#512**

SINGLE-BOARD COMPUTER

The **GMS-P5** series of Pentium-based single-board computers is suitable for applications such as medical instrumentation, data communications hardware, and automotive diagnostic equipment. The boards feature processor speeds ranging from 75 to 166 MHz, cache memory sizes ranging from 256 to 512 KB, Award BIOS, watchdog timers, and built-in IDE, floppy, and parallel and serial ports. Operating temperature range is -10" to +55°C with other options available. All boards provide PICMG compatibility for easy upgrade to Pentium technology.



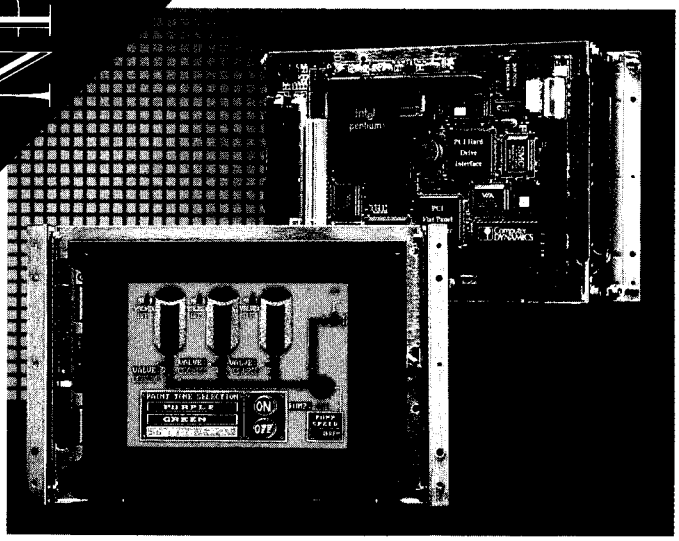
The **GMS-P5200IF** is a full-sized board with RAM up to 128 MB. Its expansion bus supports PC/104 and PCI daughterboard modules with VGA and SCSI capabilities. The **GMS-P520S** supports RAM up to 256 MB and offers a built-in SCSI-2 controller and onboard video. Its expansion bus supports PC/I 04 modules. The **GMS-P501** 1 is a half-sized board with an ISA bus and memory capability to 128 MB. Onboard video and PC/104 expansion support is standard.

Prices for OEM quantities start at approximately $350.

Granite Microsystems
10202 N. Enterprise Dr.
Mequon, WI 53092
(414) 242-8816
Fax: (4 14) 242-8825

#513

Nouveau PC



FLAT-PANEL COMPUTER

The **DisplayPac-Mini Brite** is a Pentium-based flat-panel computer that brings the advantagesandcapabilitiesof a color VGA graphics display into space-limited applications. The 6.4" TFT LCD panel has a normal luminance rating of 120 nit, but it's also available with an UltraHiBrite option that increases luminance to 450 nit and extends the backlight life to 40,000 h (4 times normal). The complete system with display, single-board Pentium PC, optional guided acoustic-wave touchscreen, and metal OEM frame measures 9.8" x 6.5" x 3.7".

The DisplayPac-MiniBrite features 4096 colors, 640 x 480 resolution and a 5.2" x 3.8" viewing area. It supports CPU speeds to 150 MHz. PCI bus is used for critical onboard functions such as video controller, Ethernet interface, and IDE hard-disk interface. Other onboard features include 512-KB cache, two serial ports and a parallel port, up to 64-MB DRAM, a floppy-drive controller, and a keyboard/mouse controller. Users can expand system functions through either the PC/I 04 or ISA interfaces. The optional guided acoustic-wave touchscreen offers the benefits of 92% optical clarity, 150-ppi resolution, z-axis for pressure-sensitive feedback, and resistancetochemicalsand scratching.

The complete system, housed in a durable easy-to-mount metal frame, draws less than 35 W and is rated at 50°C. It sells for $3880 for the 100-MHz unit with 120-nit display, 4-MB DRAM, and touchscreen ($4780 with 450-nit Ultra-HiBrite option) in quantity.

Computer Dynamics
7640 Pelham Rd.
Greenville, SC 29615
(864) 627-8800
Fax: (864) 675-O 106
sales@cdynamics.com

#514

SubVGA DISPLAY CONTROLLER

Communications and Display Systems has introduced the PC/I **04-** 1330 and PC/I 04-5000 displaycontroller boards to drive subVGA resolution graphic LCDs from a PC/104 bus. SubVGA resolution displays are popular in embedded systems where cost and space savings are essential.

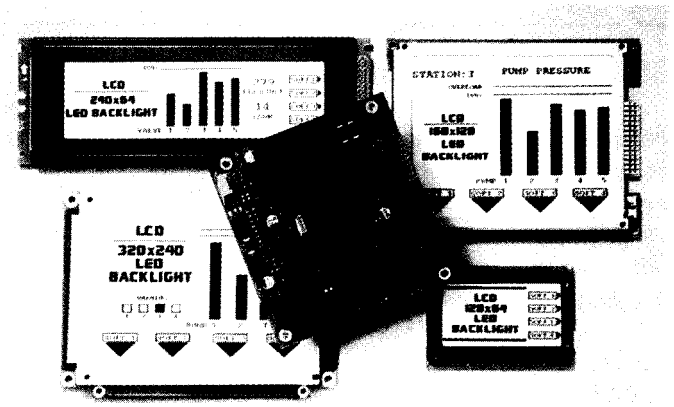
The PC/104-1330 drives most single-scan LCDs lacking onboard controllers, including CGA (640 x 200), QVGA (320 x 240) and 256 x 128, 240 x 128, 240 × 64, and 128 x 64 resolution displays. The PC/104-5000 drives most LCDs with onboard T6963C controllers, including 240 x 128, 240 x 64, and 160 x 128 resolution displays.

These controller boards include all the display contrast voltages for V_{ee}/V_{adj} plus an onboard contrast potentiometer with connection for optional customer-mounted contrast controls. The initialization circuit includes the M-signal timing circuit to safeguard the display from DC bias damage.

Prices are from $120 in quantity.

Communications and Display Systems, Inc.
194-22 Morris Ave.
Holtsville, NY 1 1742
(516) 654-l 143
Fax: (516) 654-1496

#515



Nouveau PC

NPC

ISOLATED DIGITAL I/O

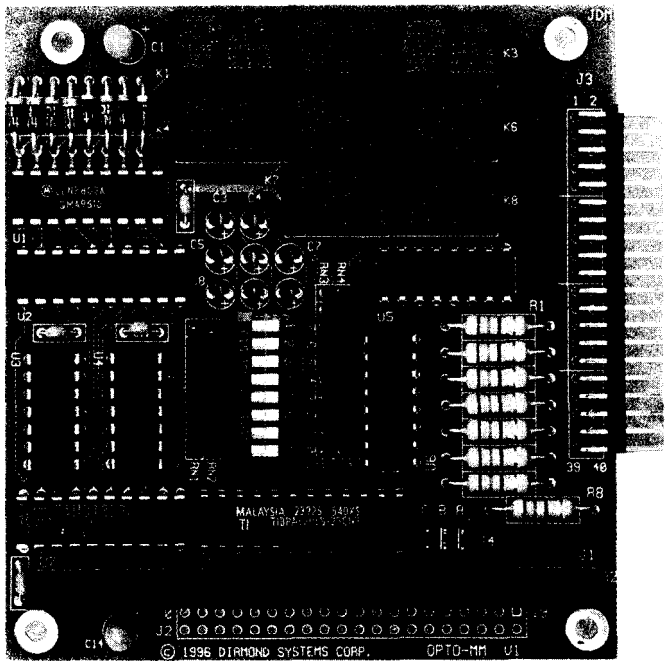
The **Opal-MM Digital I/O module** for PC/104-based embedded-computer control systems has eight optically isolated digital inputs that handle both AC and DC inputs ranging from 3 to 24 V in magnitude. In DC mode, the input is nonpolar, so the board senses the presence of an input signal regardless of its wiring polarity. In AC mode, an onboard filter circuit can be switched into each input individually to sense signals ranging from 40 Hz to 100 kHz.

Opal-MM also has eight relay outputs. Each relay is a Form C (DPDT) contact and can switch DC signals up to 30 VDC at 1 A (220 VDC at lower current) or 125 VAC at 0.2 A (inductive load). All relays maintain their power-off state on powerup to minimize potential problems. Relay lifetime is 10 million operations.

The board requires only +5 VDC for operation. All I/O signals are contained on a single 40-pin header and accessed through two 8-bit registers. Programming is simple. Prices start at $180.

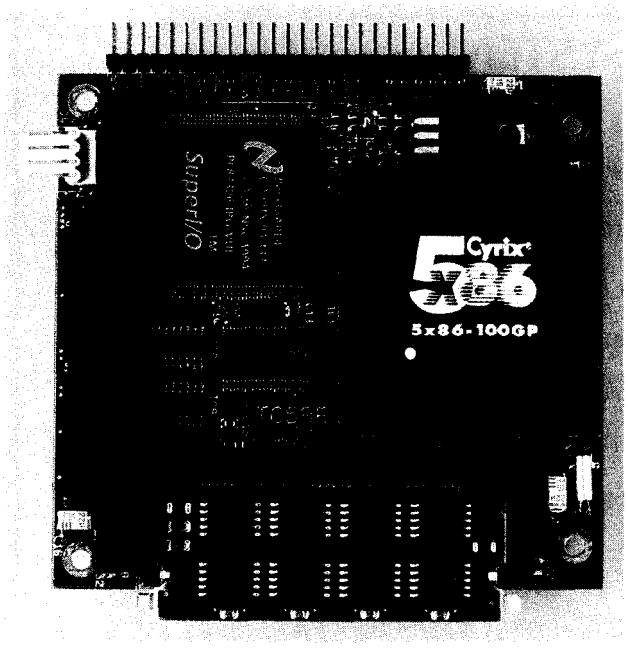
Diamond Systems Corp.
450 San Antonio Rd.
Palo Alto, CA 94306
(415) 813-1100
Fax: (415) 813-1130

#516



'586-POWERED CONTROLLER

The TC586 is a high-performance, full-function PC/I 04 module with the smallest footprint of any Pentium-class PC. It can be used as the processor module in a PC/I 04 stack or as a stand-alone single-board computer.



Running from 5 V only, the TC586 design accommodates an 80486-DX2 or DX4 '586 at 100 or 120 MHz. To keep power consumption in check, a switching regulator delivers 3.3 V to the CPU. BIOS and application code is stored in a 1-MB flash memory (solid-state disk) so users can download or upgrade onboard software. The flash disk can be bootable and have an integral flash-filing system for long-term reliability and ease of use.

Also within the PC/I 04 form factor is a DIMM socket for up to 16 MB of DRAM in a single module. Reset control circuitry ensures power-fail detection and issues clean reset pulses. Watchdog circuitry automatically restarts the TC586 if the application fails. Two RS-232 serial ports; one enhanced parallel printer port; keyboard, mouse, and sound ports; and standard PC/104 bus pins complete the I/O capability of the TC586.

A stand-alone development system TCPAK is also available that takes PC/I 04 cards and provides a floppy or mini hard disk, as well as standard connectors for serial, parallel, and VGA ports. The TC586 sells for $750 in 100 quantities.

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

#517

NouveauPC

Naren Nachiappan

Embedded PCs Go Industrial

Part 1: PCs and Automation

With lower costs and increased complexity, embedded PCs offer more modular and open solutions to industrial automation. Naren shows us how one company made the switch to a PC-based environment.

The world of industrial automation covers a large and diverse group of industries ranging from the big process industries—the four Ps of paper, petrochemicals, paint, and pharmaceuticals-at one extreme to discrete manufacturing at the other. This latter group includes automotive and semiconductor manufacturing, packaging, milling, and drilling.

Many technical and business pressures are forcing major changes in the control technologies these industries use. They are shifting from the older, highly integrated, proprietary architectures to modular, open, embedded-PC-based architectures.

In this article, I'll describe the traditional controls these industries used before reviewing the features of emerging embedded-PC-based controls. You'll see what factors are driving industries to accept PCs.

But first, let me give you some background in industrial automation and the special requirements of its applications. Then, I'll discuss traditional control technology in industrial automation.

INDUSTRIAL AUTOMATION ARCHITECTURES

In general, industrial-automation applications can be categorized by the nature of the automated process.

The spectrum ranges from continuous-control to sequential discretecontrol applications. Continuous-control applications (e.g., petroleum production) are processes that change very infrequently and are continuous in nature. Sequential control is characterized by small, discrete operation sets (e.g., milling machines).

The two major architectures in industrial automation are distributed control systems (DCSs) and programmable logic controllers (PLCs). These architectures evolved independently and were optimized to solve the different control problems posed by the two processing categories.

PLCs are microprocessor-based replacements for hard-wired relays and mechanical timers primarily used in discrete-control applications. **They** are programmed in languages resembling electrical ladder logic.

DCS evolved in industries where continuous control was the norm. DCSs are based on high-speed minicomputers controlling a large number of networked intelligent I/O devices. They typically run proprietary software, are responsible for sophisticated man-machine interfaces (MMIs), and manage large volumes of data.

Between continuous control and discrete manufacturing, there is a range of processes, typified by the food and pharmaceutical industries, where production involves a sequence of steps. These steps start an operation, add ingredients, run for a period of time, and then stop when the end product is finished.

Table 1 **roughly** outlines three segments of industrial automation according to application, technology, and response-time requirements.

It is important to note that the boundaries between PLC- and DCS-based systems are permeable. Microprocessor-based analog I/O modulescan be integrated into PLCs, so they can control continuous sys-

terns. DCSs can control PLCs at a lower level for specific discrete-control functions.

The most significant difference between continuous and discrete control is the response time of the controller device. Continuous-control applications (e.g., PID loops in the pharmaceutical industry) rarely require a response time under 100 ms. Discrete-control applications, particularly multiaxis motion-control applications, typically require a submillisecond response.

PLC-BASED CONTROL

Fundamentally, control applications monitor input data, perform application-specific transformations on the input data, and set control outputs based on the result. In its most basic form, the PLC consists of the elements necessary to implement such control applications.

Implementation	DCS	PLC	CNC
Domain	Process control	Discrete logic	Motion control
Application	Chemical Plant	Packaging	Machining
Response time	Soft (100 ms)	10 ms	< 1 ms

Table 1: The nature of the physical process dictates the response requirements of the controller. At the far end are closed-loop motion-control systems which typically require microsecond-level control.

As you can see in Figure 1, I/O is a dominant aspect of PLC-based systems. I/O modules connect the PLC to discrete input devices such as push buttons, photocells, limit switches, or analog devices (e.g., thermocouples and potentiometers). Output is directed through the I/O modules to switching closures for motors and valves.

The specific logic employed by the PLC processor for I/O transformations is typically programmed in relay ladder logic (RLL). (RLL is a language based on relay circuit design.)

EMBEDDED-PC-BASED CONTROL

A PC-based control system is depicted in Figure 2. Perhaps the two most significant differences between it and the PLC-based architecture are the software and the interface to the control elements.

A PC interface card connects directly or to a fieldbus receiving input from the plant. The interface card is mapped into the PC's shared memory space.

The control software running on the PC is hosted on an enhanced PC OS (e.g., Microsoft Windows NT). It monitors the input data in memory and manages the output data in shared memory at a maximum frequency of no more than 5 kHz.

The software's ability to maintain the desired throughput is essential in ensuring correct operation of the control system. Within discrete manufacturing, motion-control applications are at the extreme end of performance requirements. Even in this case, direct servo control is performed using a specialized DSP. The PC is responsible for a higher level of control with less-stringent performance requirements.

Device drivers, available from manufacturers like Motion Engineering and Delta Tau, download control code onto the DSP which then performs microsecond-level control of the motor.

And, timing is everything. On the plant floor, the control system is responsible for controlling the machinery. Fail-safe equipment causes a shutdown resulting in large economic loss.

In Part 2, I'll examine the specific technical details of the software and hardware enhancements which make this possible in an embedded-PC environment.

TRENDS FORCING SHIFT

The following majortechnological trends have driven the industrial-automation industry toward PC-based control systems.

Industrial PCs with functionality equivalent to a medium-size PLC now cost less than half that of the PLC ($2000 versus $5000).

Opening the factory network to standardized networking protocols is the most sweeping and powerful trend in the industrial-automation world. In the past, much of the I/O could only be connected to the controllers by the same vendor. With open I/O buses, one controller can connect to another vendor's I/O.

Two networks are prevalent today. Control-oriented field networks are inexpensive enough to connect to small discrete components. And, high-speed data networks can transfer large data files to and from the enterprise-wide network.

Ethernet is rapidly winning the standards war over MAP (Manufacturing Automation Protocol), Scramnet (from Systran), and token-ring protocols. At the field-bus level, a definite standard has yet to emerge, but industry analysts expect CAN (control area network) to dominate.

Such standardization, along with the cheap PC network interface cards, make the PC architecture a very effective platform to control plant I/O.

Most process controllers traditionally include a basic display-a bar graph of processvariablesand set-pointvalues. Push buttons adjust these values. Networked versions of these controllers can be managed from a central point where more sophisticated displays are implemented.

Requirements for operator interface (OI) are rapidly increasing to include many nontraditional functions. For example, some OI panels now include a help screen. Some sophisticated systems can walk the user through an entire sequence of operations.

For such advanced OI functions, the PC is the most suitable platform with its broad availability of PC-based video adapters and CD-ROM controllers.

Large end users are also driving vendors toward open PC-based systems. The results are already occurring in the market through specifications like Omac (open modular architecture controllers).

Omac was developed by the automotive industry to describe the preferred architecture of future controllers (i.e., ISA, PCI, or VME for bus architectures with standard interfaces between them).

To contain costs and improve maintainability of control systems, it's important to the automotive industry that PLCs are replaced by the Omac architectures. The last two major projects of GM's Powertrain group both used PCs rather than PLCs [1].

These trends are causing the PC to emerge as the next-generation control hardware of choice. Windows NT is already appearing as the OS platform for control-

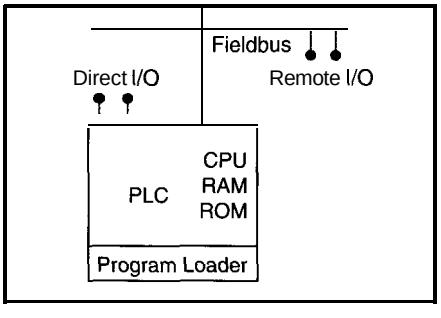


Figure 1: In PLC architecture, the program loader takes input programs, usually written in ladder logic and places it in RAM for execution by the PLC operating software. The ability to service several thousand I/O is a traditional strength of PLCs.

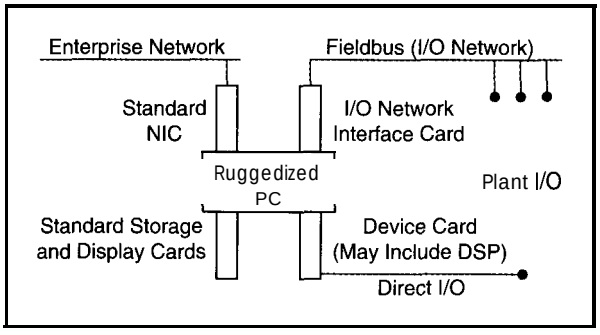


Figure 2: ***The strength of the PC-based control architecture is the ability to connect to multiple networks and use standard storage and display peripherals.***

system implementation. At ISA/95 in New Orleans, leading industrial-automation vendors, including Allen-Bradley, Fisher-Rosemount, Foxboro, GE Fanuc, Honeywell, Intellution, Opto 22, Siemens, Setpoint, Square D, and Wonderware, announced NT-based control applications [2].

Control-system implementations are also becoming more modular and are built using off-the-shelf components. This shift contrasts to the monolithic control software (particularly in DCS) of the past.

A component-software approach enables developers to rapidly build independent parts and then integrate them with other software components. Control software lends itself to a component-based approach because it's usually made up of easily identifiable independent functions (e.g., graphicdisplays, data logging, alarm monitoring, I/O scanning, etc.).

The OLE specification from Microsoft is the preferred architecture for component-based software implementation. The OLE Component Object Model (COM) lets applications connect to each other as objects.

CASE STUDY

Blandin Paper Company increased quality and saved money, down time, labor, maintenance, and training with their five-year upgrade [3]. It began by replacing old Allen-Bradley PLCs with newer ones and ended up with installing and programming PCs with ICOM software.

It started in two clay-preparation areas which make the coating that adds shine to magazine paper. This area used a batch system and clay recipes. The process accounted for about 30% of the paper cost.

The system was implemented with an operator interface and a hot backup advisor. They connected two A-B PLC-5/15s, programmed to **share** data with each other, as well as a Fisher Provox DCS on the coater and another A-B PLC-5/25 at the coater. The line clay prep had an A-B 1774 PLC with two racks of I/O and a data basic interface to a computer monitor for users.

This system had two problems. Every time a recipe changed, which was fairly regularly, the recipe program had to be virtually rewritten.

When the material-handling department wanted an inventory of ingredients used in the past 24 hours, it became evident a new control scheme was needed. So, an A-B PLC-5/30 replaced the 1774 PK.

After automating these areas, others were automated using MMI software packages from ICOM. ICOM Trend, View, and Linx software also offered advantages.

All three packages used the Allen-Bradley database, so there was no middleman database to program. These areas were upgraded using a PLC-5/25 plus the ICOM package installed on a '486DX PC.

Since the new system's performance was so impressive, thecompany purchased another '486DX and ICOM package **instead of buying another recorder and set of controllers. New** MMI screens and control schemes were designed.

At another preparation area, one of the two A-B Advisors failed and needed a card replaced. It was replaced with a clone '486DX and the ICOM software at **substantial savings. GUIs were drawn to look like** those in the Advisor, thereby reducing employee training time.

Through a process of attrition and deliberate upgrades, a network of PCs became the control system's core. Notably, the systems also worked on an A-B data highway with PLCs connected to it.

The new PC-based architecture had many benefits. Visibility into the process was dramatically enhanced, simplifying maintenance and diagnostics. A maintenance-based diagnostic MMI was created by taking the hydraulic, pneumatic, and electrical schematics and animating them.

Data sharing was easier to implement. When supervisors wanted trends available

in their offices, the Ethernet was simply extended and the data was sent through Windows Net DDE (Dynamic Data Exchange). Daily totals were also sent by DDE to inventory from the clay-prep areas.

Alarm conditions became easier to monitor and correct. Under the old system, operators had to physically attend the cookers and reset the alarm. Now, they reset the system and alarm from the control room.

The homogeneous nature of the PC enabled a single computer to backup the control PCs in several areas. A simple software-configuration switch was enough to customize it. Dial-in access to the PCs allows remote diagnostics with numerous security features, including call back.

The Windows-based ICOM software was user-friendly enough for the plant maintenance personnel to create operator interfaces and maintenance diagnostics, freeing up the plant engineering staff for more critical problems. Because all the systems have the same look and feel, training for both maintenance and the clay-preparation area staff was much easier.

Year	Units (millions)	Revenue (billions)	Growth rate (%)
1994	8.62	$4.65	
1995	9.85	$5.20	10.5
1996	11.32	$5.84	11.8
1997	12.92	$6.54	12.4
1998	14.56	$7.25	12.0
1999	16.20	$7.94	10.8
2000	17.78	$8.60	9.6
			8.3

Table 2: Worldwide ***growth in PLC control*** architectures i***s*** predicted to ***be strong*** with an increasing proportion ***of*** them ***being implemented with PC controllers.***

MARKET ALTERNATIVES

The kind of improvements experienced by Blandin are examples of some of the key factors which make the embedded PC the emerging platform of choice for control solutions. And, as Table 2 [4] shows, this market is large and growing rapidly.

Industry analysis indicates that a major shift is occurring toward the PC platform and intelligent I/O interface cards as an alternative to the traditional PLC.

Today, PC-based hardware commands only a small portion of the total revenues from the industrial-automation market. However, it is expected to grow to about 50% of the total revenues before 2000 [5].

Clearly, the impact of this market opportunity on the technical evolution of the PC platform has yet to be felt-the desktop is the driver today. However, there are several early signs that the control market is being addressed, particularly on the software front.

Several aspects of control systems place special burdens on PC software. Real-time performance, modular software construction, and embedded operation are three important requirements.

In Part 2, I'll examine how these requirements are being addressed by PC software vendors, including efforts by **Microsoft** and its partners to provide a standard Windows NT software platform for control-system vendors. EPC

Naren Nachiappan is vice president for strategic relationships at *VenturCom*. Naren initiates, develops, and *manages critical business relationships with major suppliers, customers, and partners. You may reach him at naren@vci.com.*

REFERENCES

[1] J. Yen, Supervisor of CNC/Motion Controls at General Motors' North American Manufacturing Center (Warren, MI), quoted in T. Costlow and M. Clarke, "Open Systems 'Chonge Face of Manufacturing," *EE Times*, No. 841, I-2, 1995.

[2] ISA/95 New Orleans, "Open NT winds (not Opal) blow strong," *InTech*, November, 1995.

[3] D. FergusonBlandin, Paper, Doc., http://www.isa.ord/journals/ic/digest2.htm, March, 1995.

[4] Frost and Sullivan,. "The programmable logic controller: Adopting in on environmentofchange," *InTech* 52-56, March, 1995.

[5] T. Bullock, Senior Analyst, Industrial Controls Consulting Inc., quoted in B.C. Cole, "Embedded Systems: Industrial Automation," *EE* Times, No. 858, 41, 1995.

SOURCES

Windows NT
Microsoft Corp.
One Microsoft Way
Redmond, WA 98052
(206) 882-8080
Fax: (206) 936.7329

Windows NT Development Suite
VenturCom
2 15 First St.
Cambridge, MA 02 142
(617) 661-1230
Fax: (617) 577-I 605
E-mail: mds@vci.com

IRS
425 Very Useful
426 Moderately Useful
427 Not Useful

Mike Baylis

Embedding QNX in Flash

Want to do real-time data *acquisltion* with a remote diskless embedded PC? If so, check out what Mike has to say about embedding a real-time operating system in flash. It might be just what the doctor ordered.

Using solid-state disks and real-time operating systems (RTOS) in embedded applications aren't new concepts. However, flash memory opens up new possibilities for embedding an RTOS and supporting a file system that is in-system updatable.

In this article, let's explore flash solid-state disks and choose an RTOS that supports these devices. I'll embed the RTOS in a flash solid-state disk and discuss possible real-world uses of this combination.

WHY A REAL-TIME OS?

Many users of embedded-PCs use DOS or proprietary OSs with their application running on top as an EX E file. But, there **are** some applications where DOS just won't do. DOS doesn't support multitasking, it isn't reentrant, and it isn't scalable.

True, there are multitasking libraries and executives that manage tasks and work with DOS. But, if you also need to manage peripherals such as video, keyboard, disks, and network interfaces, you're probably better off with an RTOS.

For a solid overview of factors to consider when choosing real-time kernels and operating systems, see Rick Lehrbaum's "The Soft Side of PC/ 104" (INK 69).

WHY QNX?

QNX is an excellent choice for an RTOS that is embeddable and fully supports flash solid-state disks (FSSDs). It's a POSIX-compliant real-time multitasking OS based on a small (10 KB) microkernel that communicates with cooperating processes via message passing.

Its interprocesscommunication and preemptive priority-driven scheduling makes it ideal for true real-time uses. It is also scalable. Just add or subtract processes.

Since higher-level QNX OS services (e.g., file systems, console I/O, and networking) are provided by processes, you only include the ones you need.

This modular architecture makes QNX ideal for scaled-down embedded applications with limited storage. And, it makes it possible to use a POSIX-standard API.

SSD FUNDAMENTALS

SSDs use ROM, SRAM, or flash memory to gain the same nonvolatile storage capabilities as rotating magnetic media. In applications where size, power consumption, and/or shock and vibration forbid the use of disk drives, SSDs excel.

In addition to the memory, SSDs must also interface to the host CPU via the system bus (e.g., memory cards and onboard flash arrays) or a PCMCIA-ATA interface (PC cards).

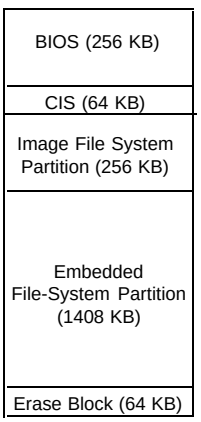


Figure *1: The logical organization of a 2-MB* QNX *solid-state disk includes partitions for an image and an embedded file system.*

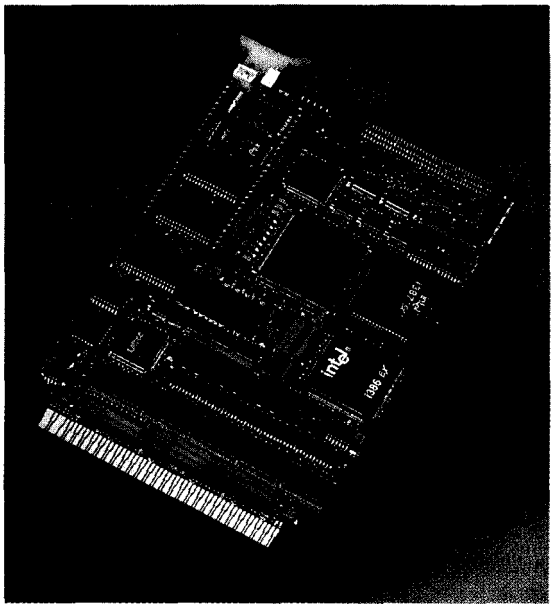


Photo 1: A 4.5" × 6.5" '386EX computer provides a platform for the QNX real-time operating system. For real-time applications where DOS and rotating disk media won't do, QNX can be embedded in flash solid-state disks on embedded systems like this.

And, there's the software driver. It enables the OS and applications to access files in the SSD. The driver is either loaded during system initialization as an installable driver or baked into the BIOS. The BIOS method is most common for bootable SSDs.

Of the three device families, flash memory is becoming more popular. Since it's the lowest cost, nonvolatile read/write semiconductor storage available today, it is the preferred choice for SSDs.

Flash memory is available in densities comparable to ROM, and unlike RAM, it has no refresh or battery requirements. Its fast access times frequently eliminate the need to shadow code in RAM. Intel, AMD, and Atmel are its top three suppliers.

Three common architectural flavors of flash devices are bulk erase, block erase, and boot block. Bulk-erase devices such as Intel's 28F010 256K x 8 flash memory must be totally erased and reprogrammed to update any portion of its contents.

Concerns about a reset or power failure occurring in the middle of reprogramming these devices led to the development of block-erase flash devices. These devices are logically divided into erase blocks. Portions of the device reprogram without altering the contents in other portions.

Intel's 28F008 1 M x 8 flash device is a popular example. It has 1 -MB storage and isdivided into 16 x 64Kerase blocks, each having 100,000 erase cycles.

Boot-block devices can have a portion of their erase blocks hardware-or software-locked to protect critical boot and download code.

LOGICAL ORGANIZATION OF FSSD

FSSD interfaces (i.e., sockets) subdivide into regions of similar memory devices (i.e., RAM, ROM, or flash). They can be logically subdivided into partitions for an image file system where the boot image resides and for an optional file system for storing directories and files.

A data block called the Card Information Structure (CIS) stores the memory-device types found within a region and how the FSSD is partitioned. For an FSSD to boot, it must include a boot loader that is loaded and executed by the system's BIOS, a CIS describing the image-file-system partition, and a boot image.

Since flash-memory devices are typically subdivided into erase blocks, a spare block is reserved for space reclamation. Figure 1 represents a 2-MB SSD on a ZT 8904 '386EX CPU board consisting of two onboard Intel 28F008 1 M x 8 flash devices.

QNX SSD SUPPORT

As an extension to the QNX 4.22 OS, QNX's embedded kit supports booting QNX from a SSD, the QNX file system in a SSD, and execute-in-place (XIP).

The heart of this kit is the embedded file-system manager which functions as the file manager and the SSD device driver based on the PCMCIA 2.01 socket services specification. It has several vendor-dependent versions for onboard flash array SSDs, as well as a generic version to support Intel's 82365SL PCMCIA controller device.

Also included are hardware-independent utilities for erasing and formatting SSD partitions, creating embeddable QNX boot images, and compressing files. Although QNX supports SSDs based on all three device types, I'll focus on FSSDs.

Listing 1: This file describes the regions and partitions of Ziatech's 2-MB solid-state disk. It's used to create the Card Information Structure (CIS) for the SSD.

```
region
  size 1792k
  inter1 eave 1
  jedec 0x89a2
  boot-file "/boot/sys/boot.zt8904-2M"
  partition
    type image
    size 256k
    attribute xip_align
  partition
    type efs
    size 128k
    attribute largest erase-align write-align
```

ROLLING A QNX FSSD

Let's initialize and partition an FSSD, embed a QNX OS image, and copy files to the file-system partition of an FSSD.

The first step is to start the embedded file-system manager specifying the JEDEC identifier, size, block, and offset of the SSD. But, what's a JEDEC identifier?

Most flash devices have a manufacturer and device ID programmed into them that's read via software. SSD drivers read this identifier since device manufacturers use different programming algorithms. The driver automatically places itself in the background and creates a special device file for raw access to the SSD and the embedded file system.

The efsinit utility specifying the device file erases the SSD and clears its contents. The utility can also format file-system partitions once they're created.

mkcis reads .info which describes the SSD socket's memory regions and partitions and writes a CIS to the SSD. This enables defined partitions to be accessed.

The next time the embedded file-system manager starts, the driver gets the JEDEC identifier, size, block, and offset from the CIS. If the SSD is bootable, .info must specify the boot loader. Listing 1 shows the CIS.info file for a 2-MB FSSD on a Ziatech ZT 8904 board with image- and embedded-file-system partitions.

xip_align, erase-align, and write_align specify boundaries within the SSD that the partitions must be aligned on (i.e., 4K boundary for image partitions and erase and write-block boundaries for embedded-file-system partitions). The largest attribute specifies that the partition should take up the largest free area in the region.

The QNX OS generates image files with buildqnx. However, these images can't be embedded since the supplied boot loaders need boot images in a special format. romqnx converts these images into boot images. If the SSD is ROM or flash and linearly mapped in the processor's address space, XIP can be used. The code portion of each module is then executed directly out of ROM or flash instead of being copied to RAM first. Data modules are always copied to RAM, so if desired, the utility can compress these into the SSD.

mkimage formats the image-file-system partition and **optionally** copies the romqnx boot image to the partition.

Copy the application and associated processes or files to the embedded-file-system partition. efsinit specifies the device file for the file-system partition.

The QNX cp utility copies files from a disk, or optionally, the cbpe utility copies and compresses these files using a byte-pair encoding algorithm.

A WORKING EXAMPLE

Now let's build a QNX bootable system that supports keyboard, screen, and user login. I'll use the Ziatech ZT 8904 STD bus CPU board with twoonboard Intel 28F008 1 M x 8 flash devices (see Photo 1).

The devices are memory mapped at Ox 1400000-0x1 4fffffh and Ox 1300000–0x1 3fffffh, respectively, with the upper 256 KB of the first flash device reserved for the BIOS. For video and keyboard support, I'll add a zVID2 Local bus video daughterboard.

I assume QNX 4.22 and the embedded kit are on an IDE hard disk in an STD bus card cage and that the ZT 8904 is booting from this disk. When I'm done, I'll remove the hard disk and boot from the FSSD.

Listing 2 shows Makefile, which initializes and formats the SSD, creates the CIS, builds an embeddable QNX boot image, and copies the required files to the embedded-file-system partition. Before running Makefile, start the embedded file-

listing 2: This sample *Make f i 7 e* builds a bootable **QNX solid-state** disk supporting screen, keyboard, and user **login** for a **ZT 8904 '386EX STD32 board** with **2-MB** flash. Even though **the SSD consists of two 1**-MB devices, **the** driver **makes it** appear as one contiguous **socket.**

```
# Makefile for ZT 8904 SSD (2M)
NODE            1
TARGET-NODE = 1    # node number you are building for
FLASH-RAW     = //$(NODE)/dev/skt1
FLASH-IMAGE = //$(NODE)/dev/sktlimg1
FLASH_EFS     = //$(NODE)/dev/efslp1
BOOT_IMAGES = lbootlsys
BOOTFILE      = $(BOOT_IMAGES)/boot.zt8904-2M

flash:  zt8904.info.2M $(BOOTFILE) os.flash.xip.2M
  efsinit $(FLASH_RAW)
  mkcis -o $(FLASH_RAW) zt8904.info.2M
  cat os.flash.xip.2M > $(FLASH_IMAGE)
  efsinit $(FLASH_EFS)
  mkdir $(FLASH_EFS)/bin
  mkdir $(FLASH_EFS)/etc
  mkdir $(FLASH_EFS)/etc/config
  cbpe -f /bin/login $(FLASH_EFS)/bin/login
  cbpe -f /bin/sh $(FLASH_EFS)/bin/sh
  cbpe -f /bin/tinit $(FLASH_EFS)/bin/tinit
  cbpe -f /bin/Dev $(FLASH_EFS)/bin/Dev
  cbpe -f /bin/Dev.con $(FLASH_EFS)/bin/Dev.con
  cp -t letclconfiglsysinit $(FLASH_EFS)/etc/config
  cp -t /etc/passwd/etc/shadow $(FLASH_EFS)/etc

os.flash.xip.2M: os.flash.build.2M $(BOOTFILE) Makefile2M
  buildqnx -v n=$(TARGET_NODE) os.flash.build.2M tmp
  romqnx -c -d0 -R0x1305000 tmp tmp2
  mkimage -o os.flash.xip.2M tmp2
```

system manager for the ZT 8904 and specify a JEDEC identifier of 89a2, a size of 1792 KB, and a 64-KB erase block.

Now reboot the system after removing the IDE drive/controller card and change the boot device in CMOS setup from fixed to special. The ZT 8904 reboots from the onboard SSD and displays the QNX boot loader message and a login prompt.

LET'S GET REAL

But, what are the practical uses of RTOSs using FSSDs?

Suppose you need a diskless remote RTOS data-acquisition system which networks with a host for application files but needs to self-boot lest the host be down.

Also, the host needs to download parameter files into the remote's FSSD. If the network is down, the remote uses the last host-supplied parameter file.

QNX works well in this scenario since networking is built into its OS. You don't need a network OS on top of an existing OS (like DOS requires).

And, there you have it. A true real-time embedded application with system-updatable file-system support, no rotating disks, and an RTOS with an FSSD. EPC

Mike *Baylis* is *an applications engineer for* Ziatech, a *manufacturer of small embedded computers for STD32 and CompactPCI Bus. You may reach him at* mike_baylis@ziatech.com.

SOURCES

AMD
5204 E. Ben White Blvd
Mail Stop 604
Austin, TX 78741
(512) 602-5651
Fax: (512) 602.7639

Atmel Corp.
2125 O'Nel Dr.
San Jose, CA 95131
(408) 441-0311
Fax: (408) 4364200

28F010, 28F008, 82365SL
Intel Corp.
Literature Center
P.O. Box **7641**
Mt. Prospect, 1160056
(916) 356-7368
Fax: (847) 296.3699

QNX 4.22 OS, Embedded Kit
QNX
175 Terence **Matthews** Cres.
Kanata, ON
Canada K2M **1 W8**
(613) 591-0931
Fox: (613) 591-3579

ZT 8904 386EX
Ziatech Corp.
1050 Southwood Dr.
Son Luis **Obispo, CA 93401**
(805) 541.0488
Fax: (805) 541.5088
BBS: (805) 541-82 **18**

IRS

428 Very Useful
429 Moderately Useful
430 Not Useful

Rick Lehrbaum

Where No '104 Module Has Gone Before

The Advent of PCI-PC/104

By far, the ISA bus has been the most stagnant aspect to PC evolution. While CPU speed, integration, and real estate have progressed, ISA with all its problems has remained a bottleneck. With PCI, PC buses take a leap ahead.

In case you hadn't noticed, desktop PCs are undergoing a major lifecycle transition. A new exotic virus is attacking PC after PC. The steady onslaught seems unstoppable, and no PC-desktop, laptop, or embedded-appears safe from itseventual domination. We know this relentlessvirus by a three-letter acronym-PCI.

Today, it's unlikely you'd buy a PC without PCI-whether for yourself, a friend, or a business. Why?

A WALK THROUGH HISTORY

Run a benchmark like Norton SY S I N FO and notice how the CPU performance of your Pentium machine compares to that of the original 4.77-MHz 8088-based PC. You'll probably discover that your Pentium system clocks between 300 and 400 times faster than the 8088 of fifteen years ago.

However, although the PC's CPU speed has increased dramatically, its bus bandwidth has remained unchanged. The 64-bit Pentium transfers data at 33 megawords per second, which translates to theoretical data rates of up to 264 MBps.

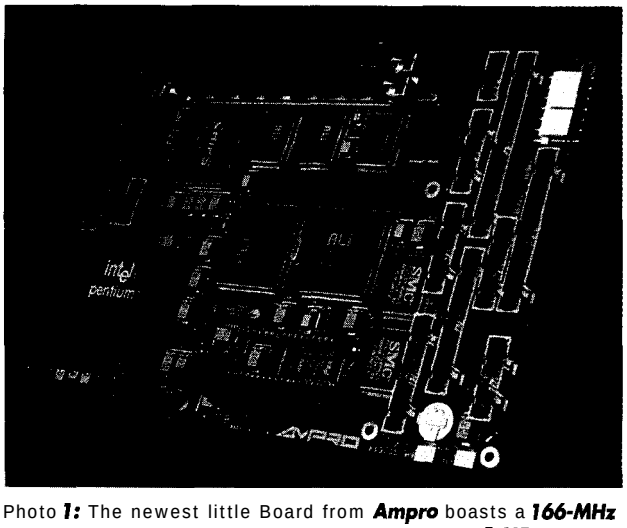


Photo 1: The newest little Board from **Ampro** boasts a **166-MHz** pentium CPU. With that kind of horsepower, the **5-MBps** speed limit of **PC/104** might become a real bottleneck in many application+. The solution--add PCI. Can you spot the **120-pin PCI** connector that **Ampro** added to **PC/104?**

On the other hand, the PC bus has a maximum practical throughput rate of ~5 MBps. Seems like a problem, doesn't it?

To overcome the PC-bus bottleneck, motherboard manufacturers have tried two approaches. For one, they migrated high data-rate functions (e.g., video and hard-disk interfaces) from bus expansion cards directly onto the motherboard.

But, if the video interface is built into the motherboard, you get no choice in features, and you lose the ability to alter or upgrade it. What you gain in performance, you sacrifice in modularity and flexibility. The same is true for built-in diskdrive interfaces.

Another attempt at fixing the bus bottleneck was to add Local bus connectors to the motherboard's add-in card slots. The added connector-VLB (VESA Local Bus)-carried certain motherboard internaladdress and data bus signals. VLB provided

PCI

a high-speed data interface between the CPU and peripherals. But, since VLB's data and control signals were based on the Intel '486DX CPU, future Intel CPUs with differing data and control signals wouldn't support VLB.

UPPING THE MAXIMUM SPEED

Intel knew that 'x86 CPU revenues were highly dependent on the PC's continued popularity and success. With CPU performance multiplied by orders of magnitude, something had to be done about the PC bus bottleneck. So, Intel created PCI.

According to plan, PCI offers a high-speed virtualized interface (i.e., standardized bus). It interfaces CPUs of many types to various system functions.

To promote PCI, Intel established a PCI Special Interest Group. Their efforts have been extremely successful! No microcomputer bus standard has spread so rapidly.

A DRIVING LESSON

Given the performance evolution of the Intel 'x86 CPU family, it was only a matter of time before the PC needed a faster bus. There's a limit to how long you can tolerate 5-MPH speed limits!

It's one thing if your buggy has a horse pulling it. But, if you're driving a 350-hp turbocharged sports car, it's hard to obey speed-limit signs. Intel offers to raise the limit to 132 MPH. Interested? You bet!

Actually, the desktop-PC industry didn't accept PCI right away. They hesitated because the '486DX CPU still dominated the desktop when Intel began preaching about PCI. And by nature, VLB interfaces neatly (and inexpensively) to a '486DX.

Also, there was a cost penalty associated with PCI because it's more generalized in function. So, in the desktop-PC market-where the three main buying criteria are price, price, and price-acceptance of the higher-priced PCI bus was not instantaneous.

While the computer rags blazed with headlines like "Which bus will prevail—VLB or PCI?", anyone bothering to ask a computer architect quickly learned that it was only a matter of time until PCI won.

All that was needed was a change in how the CPU talked to the chips around it. The Pentium brought this change. Its interface signals are quite different from those of the '486DX. The Pentium killed VLB dead!

Now, PCI is appearing on virtually every desktop PC.

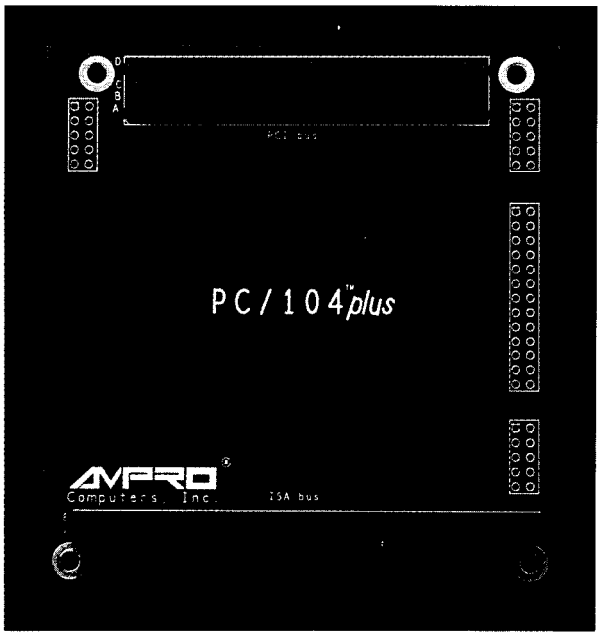


photo 2: The bottom connector is the usual **PC/104** ISA bus. The fop connector is **the** new **120-pin** self-stacking PCI bus.

OFF-ROAD VEHICLES

But what about off the desktop—in the embedded systems world?

There are lots of potential benefits to using a Pentium in embedded applications. Code runs much faster. Computational results come more quickly. Decisions happen sooner.

With a Pentium-based embedded PC, you can implement performance-sensitive real-time closed-loop feedback systems. Manipulate images. Process vast arrays of data. The possibilities seem endless!

But, what about the PC bus bottleneck? With a Pentium CPU in that embedded system, you run into the same problem you had on the desktop.

It's not long before you discover your beloved embedded-PC bus interconnect—PC/I 04—can't keep up with your need for high-speed bus transfers.

WHITHER PC/I 04?

You already know that PC/ 1 O&based embedded systems come in lots of shapes and sizes. The CPU can be PC/I 04 form factor, or it may be a PC/I 04-expandable single-board computer (SBC). Or, it might take the shape of an ISA expansion card, plugged into a multislot passive backplane.

Whatever its shape, embedded-PC performance has, until recently, been limited to a '486DX CPU or less. In typical data-acquisition, system-control, and operator-interface applications where embedded PCs have been useful, PC/I 04's 5-MBps bus throughput rate

	Desktop PCI	Passive Backplane PCI	PMC	CompactPCI	CardBus	Small-PCI	PC/104-*Plus*
Dimensions (in.)	12.3 × 3.9 (long) 6.9 × 3.9 (short)	12.3 × 3.9	5.9 × 2.9	6.3 × 3.9	3.4 × 2.1	3.4 x 2.1	3.8 × 3.6
Area (in?)	48 (long) 24 (short)	48	17	25	7	7	13
Bus Connector	edge-card	edge-card	pin & socket	pin & socket	pin & socket	pin & socket	pin & socket
Includes ISA	no	yes	no	no	no	no	yes
Installation Plane	perpendicular	perpendicular	parallel	perpendicular	parallel	parallel	parallel
Self-Stacking	no	no	no	no	no	no	yes
Positive Retention	no	no	yes	yes	no	no	yes
Standards Body	PCI-SIG	PICMG	IEEE	PICMG	PCMCIA	PCI-SIG	PC/104
Primary Application Area	Desktop: motherboard expansion	Industrial: backplane expansion	Industrial: VME mezzanine	Industrial: backplane expansion	Laptop: end-user additions	Laptop: factory options	Embedded: SBC expansion

Table I: PCI is mutating into many shapes and sizes suited to o variety of purposes. This table compares seven available versions. Because **PC/104-Plus** is compact, self-stacking, and includes both the ISA and PCI, it's a great choice for the backbone of *tomorrow's* **performance-critical** embedded-PC systems.

was considered fast enough. But, with the Pentium, the bus bottleneck becomes a serious problem.

"You're kidding," you say. "A Pentium on PC/I 04?" A standard PGA-packaged Pentium is too big to fit on a PC/I 04 form-factor CPU module, right?

Sure. But, don't forget about the larger sized PC/I 04-expandable SBCs. Take a lookatthe new Ampro Little Board/P5*i* (see Photo 1). In designing this new Pentium-based SBC, Ampro engineers had to support the high-bus bandwidth requirements of performance-oriented applications.

With the advent of the Pentium-based embedded-PC, PC/104 had clearly arrived at a fork in the road. It was faced with a simple choice-adapt to the future or be relegated to the scrap heap of history.

The solution-add PCI to PC/I 04.

WE'VE COME A LONG WAY, BABY

Remember, PC/104 is basically the desktop-PC architecture repackaged to meet the specialized needs of embedded systems. Technology trends of the desktop eventually find their way onto PC/104. The desktop PCI phenomenon was destined to come to PC/I 04.

But, what form would it take?

As you can see in Table 1, there are a number of groups adapting PCI to various buses and form factors-PMC, PICMG, CompactPCI, SPCI, and CardBus. Despite the seemingly endless list of buses, none of the PCI variants quite made it since Ampro wanted to retain PC/I 04 strengths:

- is compact (3.6" x 3.8")
- is self-stacking (expands without backplanes or card cages)
- has a pin-and-socket bus connector (highly reliable in harsh environments)
- uses four corner mounting holes (resists shock and vibration)
- offers low powerconsumption (low power requirement and heat generation)
- is fully PC compatible

These features have made PC/104 the choice of thousands of embedded-system developers worldwide during the past five years. To be really useful, an embedded PCI bus for Pentium-based embedded PCs should preserve these benefits.

Oneadditional requirementwould make the new embedded-PC1 bus especially flexible. If it could be stacked and used along

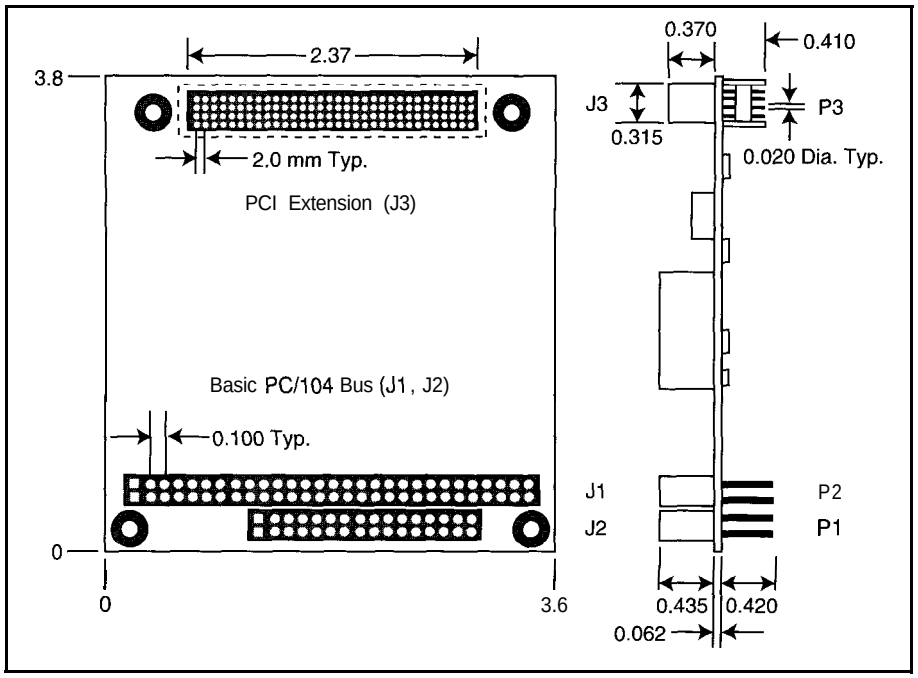


Figure 1:* PC/104-Plus** modules are compatible ***with the normal PC/104 format, except ***that a 120-pin high-density PCI connector was added. Unlike*** the exposed ***pins of PC/104's ISA connectors (P1/P2), the pins of the PC/104-Plus PCI connector (P3) are strengthened and*** protected by a plastic ***shroud.***

with standard stacked PC/104 modules, we'd have something worth using.

Finding a self-stacking PCI bus connector that was backwards compatible with PC/104 presented a real challenge. An exhaustive search for an off-the-shelf connector turned up nothing.

Fortunately, Ampro uncovered a willing ally in connector supplier (and PC/104 Consortium member) Samtec. Together, they defined a new 120-pin high-density 2-mm connector with "stackthrough" pins of a length that exactly matched the existing bus connectors of PC/I 04.

Also, they developed a special plastic shroud with two important functions. It guides the male portion of one connector as it mates with the female portion of the next connector in the stack. And, it protects the male connector pins, which are somewhat thinner and more vulnerable than the normal PC/ 104 (ISA) connector.

The result, a merger of PC/104 and PCI, has been dubbed PC/104-*Plus*. It satisfies every one of PC/I 04's objectives.

PC/104-*PLUS* ON THE WAY

Photo 2 and Figure 1 show what Ampro came up with. Notice the overall module dimensions are exactly the same as those of standard PC/I 04.

The new PCI bus fits comfortably between the two PC/I 04 mounting holes on the edge of the module opposite the regular PC/I 04 bus. For clarity, I'll call the old PC/I 04 bus connector pair "104-pin ISA" and the new connector "120-pin PCI."

Figure 2 illustrates a typical PC/104-Plus stack. As you can see, you can combine PC/ 104 modules (8 and 16 bit) with PC/I 04-*Plus* modules (32 bit) in the same stack, as long as modules of the same kind are next to each other.

Here are some of the specs of PC/ 104-Plus modules:

- spacing between stacked modules: 0.6" (same as PC/I 04)
- data throughput (max.): 132 MBps
- bus drive current (min.): 3 mA (most signals)
- bus load current (max.): 700 µA
- number of PCI modules per stack (max.): 5 (including CPU module]

Although PC/I 04-*Plus* modules have connectors for both ISA and PCI buses, only the PCI bus normally interfaces with components located on the module. The 104-pin ISA bus on a PC/I 04-*Plus* module passively transfers ISA signals through to the next module in the stack.

Who knows? Some day, when ISA is long gone, PC/ 104-*Plus* might lose its 104-pin ISA connector pair entirely! (Would we still call them/I 04 modules?)

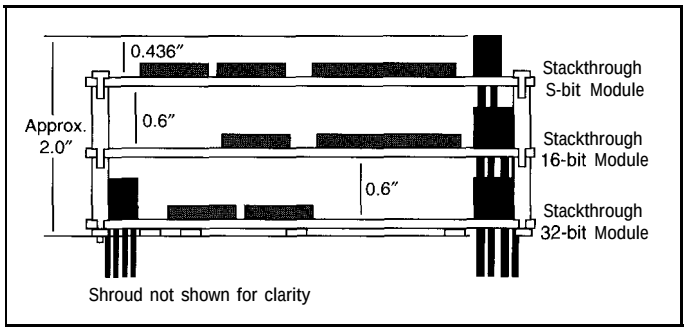


Figure 2: ***PC/104-Plus* retains the *stacking* flexibility *of PC/104.*** You can combine ***8-*** and ***16-*** bit ***PC/104*** modules with ***32-bit PC/104-Plus (PCI)*** modules as long as you keep similar bus ***types*** next to ***each*** other. Four spacers ***rigidly*** attach ***the modules*** to ***each*** other.

Table 1 compares PC/I 04-*Plus* with six other variants of PCI, including normal desktop PCI. Notice how each implementation of PCI is best suited to a slightly different purpose.

I like to think of PCI as an architecture, not just a bus-like the PC architecture itself. PC/I 04-*Plus*, like PC/I 04, is simply a repackaged version of PCI tuned to the special needs of modular embedded systems where space is scarce and ruggedness is paramount.

WHERE DO WE GO FROM HERE?

You're probably wondering, "What's the status of PC/ 104-*Plus* as far as becoming an official standard?"

Ampro intends to make this new PCI-enhanced version of PC/104 an open, public-domain standard. Expect it to be availablewithout restriction or license fees, just like the PC/104 standard, and most likely through the PC/I 04 Consortium.

Around a dozen companies are already developing PC/I 04-*Plus* products. Some are off-the-shelf modules intended for sale to other companies using PC/104-Plus within embedded systems (e.g., video frame grabbers, 1 00BaseT Ethernet LANs, and high-speed analog and digital I/O interfaces). Other PC/I 04-*Plus* developments in process are proprietary circuit boards for specific embedded-system projects.

Ampro has generated a detailed PC/I 04-*Plus* "proposed specification" and has submitted it to the PC/I 04 Consortium Board of Directors. They have recommended it be adopted as a PC/I 04 Consortium-sponsored standard. During the new spec's evaluation period, the only way to get a copy is request one from Ampro.

Like PC/I 04, the PC/I 04-*Plus* spec is essentially a mechanical definition only. It shows how to repackage PCI onto a PC/I 04 form-factor stackable module. Like the PC/I 04 spec, PC/I 04-*Plus* doesn't include detailed information on signal functions or timing.

Therefore, if you need detailed technical information about the PCI bus itself, you'll have to consult another reference on PCI (e.g., the PCI Local Bus specification). There are also some good books about PCI from Annabooks and Computer Literacy Bookshops.

So, read up and get ready for PCI's assault! Not only will it be showing up on your desktop soon-if it hasn't already-it will be invading your embedded projects, too! PCQ.EPC

Rick lehrbaum cofounded Ampro Computers where he served *as vice president of engineering from 1983* to 199 1. *Now,* in addition to *his duties as Ampro's vice president of strategic development, Rick chairs the PC/ 1 04 Consortium. He may be reached at rlehrbaum@ampro.com.*

SOURCES

Little **Board/P5*i*,PC/104-*Plus* Spec**
Ampro Computers, Inc.
990 Almanor Ave.
Sunnyvale, CA 94086
(408) 522-2 100
Fax: (408) 720-I 305
info@ampro.com

PCI Local Bus **Spec**
PCI Special Interest Group
2575 NE Kothryn, Ste. 17
Hillsboro, OR 97124
(503) 693-6232
Fax: (503) 693.3444

PCI books
Annabooks
1 1838 Bernardo Plaza Ct., Ste. 102
San Diego, CT 92 128-24 14
(619) 673-0870
Fax: (619) 673-I 432

Computer Literacy Bookshop
2590 N. First St.
Son Jose, CA 95 13 1
(408) 435-0744
Fox: (408) 435-I 823

IRS

43 1 Very Useful
432 Moderately Useful
433 Not Useful

Applied PCs

Fred Eady

The Black Art of Embedded Peripherals

Fred brews *a* pot **of** *embedded-PC* peripherals, *and* we're invited *to* the feasf. **He uses Intel's** *ApBuilder* **and** *RadiSys's EXPLR1 to demonstrate the rich* **set** *of* on-chip peripherals in *Intel's* '386EX.

Witchcraft. That'swhat this embedded-PC stuff is. In Salem, I'd have been unquestionably burned at the stake! Here's why.

When preparing an *EPC* article, I work with four full-blown desktop PCs, a couple of laptops, and of course, the one small-footprint embedded PC featured. I use voice, fax, and Internet to communicate with my editors and the featured vendors.

These dark instruments of communication and subservient computers (especially the miniature embedded beasties) would be perceived as my cat, kettle, and broom.

I'd face my inquisitioner mumbling "embedded computer" or "BIOS." I'd beguilty, guilty, guilty. My inquisitioner would have no problem noting the evidence.

I make inanimate objects move (using embedded robotics applications). I speak regularly to invisible supernatural spirits via my looking glass (i.e., the Internet).

In the cover of darkness, I concoct gnarly and sometimes smelly potions (writing and debugging embedded programs). And, from seemingly nowhere, I create smoke, fire, and blinding light (from the occasional embedded hardware snafu).

Shucks, according to some upstanding citizens of my community, I've been known to turn into an animal. My friends tell me I do this at full moon cycles. And, what is the moon doing as I write this? Go figure.

They'd have to burn me to a crisp to keep me from stirring this pot of embedded-PC peripherals. So, I'll just wiggle my nose and conjure up some tasty peripheral stew.

A STAKE IN THE GROUND

If you think about it, embedded platforms put the maximum amount of computational and peripheral power into a single integrated circuit. It eliminates the need for the external parts and firmware present in a comparable nonembedded environment.

The immediate and obvious advantage of embedded platforms is that power and space are conserved. A perfect example is Intel's '386EX. The '386EX processor has a modular, fully static Intel '386SX CPU with a System Management Mode (SMM) for enhanced power management.

The '386EX processor has a 16-bit data bus and a 26-bit address bus, supporting up to 64 MB of memory address space and 64 KB of I/O address space. The real carrot-the rich set of on-chip peripherals.

At this particular moon phase, I have three embedded PCs in the shop, all using the '386EX. So, guess which processor's peripherals you'll be reading about.

GET ON THE BROOM!

Like you, I've read technical journals for years. We both know block diagrams and graphics are essential to better understand any hardware or software concept.

So, using some of this dark technology, I'll present code and pictures of the stuff that makes up Intel's '386EX peripheral set.

Although I chose a particular processor core, the feel of these peripherals is like other micros in its class. Most 'EX peripherals have roots in the beginning of PC time.

To me, the graphical approach is more logical than manually assimilating and collating all the data and associated diagrams in the Intel '386EXb data books.

So, I'll show you Intel's EXPLRI evaluation platform and ApBuilder. You'll find it more interesting than block diagrams.

WITCH'S BREW

ApBuilder is an evaluation tool that provides online reference tools and programming aids for the '386EX. I got my copy with an EXPLRI evaluation kit, but you can download it from Intel's BBS or website. Photo 1 is the ApBuilder intro screen.

The great thing about ApBuilder is that we can manipulate the '386EX peripheral set interactively.

ApBuilder also generates code to control the peripheral functions. Select the characteristics of the peripheral you're working with, and ApBuilder builds a code snippet for your application source code.

The '386EX embedded-microprocessor user's manual is exactly 1.332" thick while ApBuilder is easy to use, interactive, and graphic intensive. Which to use?

A show of hands for the book? I don't see a single hand, so ApBuilder it is! Good choice. You'll see it doesn't have to be difficult to be embedded.

THE KETTLE

EXPLRI complements ApBuilder application software perfectly. I'm using the RadiSys variant of the **EXPLR1**.

It is based on Intel's '386EX embedded microprocessor and is surrounded with special RadiSys hardware (see Photo 1 in Brad Reed's excellent offering in *INK* 71, "Rolling Your Own Intel '386EX-based Embedded PC").

Although it's missing a few interrupts and really isn't a full-blown '386 embedded system, I'm approaching EXPLRI as a standard '386 embedded platform.

THE STEW

Most hardware is useless without software. EXPLRI comes preloaded with a demo version of Annabooks' ROMable DOS. Unfortunately, this demo isn't robust enough for us. It's nice to look at, but it won't open up **any** of the 'EX internals to us. I'm sure the full-blown version of ROMable DOS would be useful, but I don't have it.

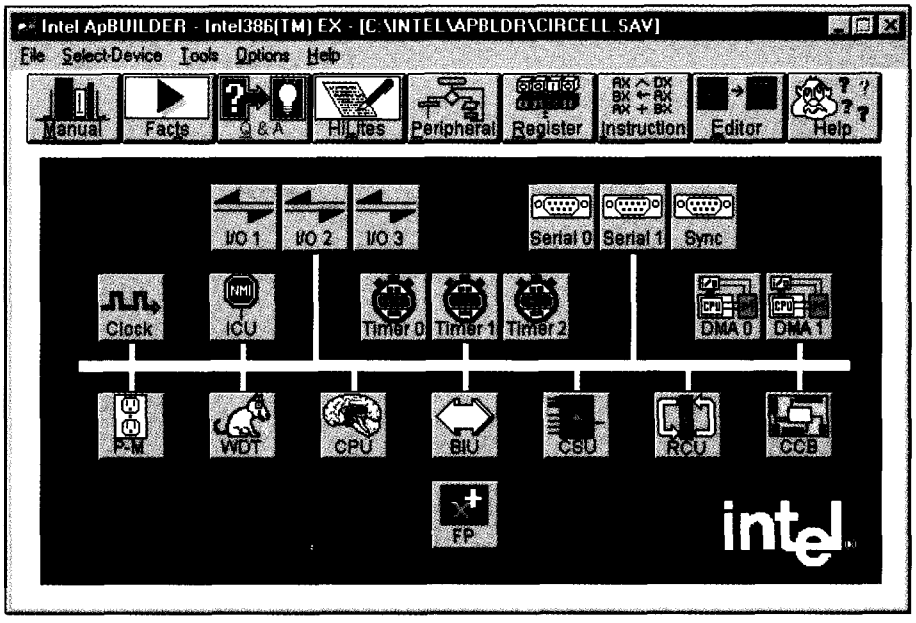


photo 1: Every microprocessor on the market should have *its own version of* **ApBuilder.** Just about everything you need to know about *the* '386EX *is* behind the buttons.

EXPLRI is also shipped with a developmental BIOS, PhoenixPICO, provided by Phoenix Technologies. The good news is that the BIOS initializes the EXPLRI hardware so I can introduce an embedded monitor system from Phar tap Software, the TNT Embedded ToolSuite.

As its name implies, the toolsuite is dynamite! I'll use the ETS software to enable some of the software snippets I generate using ApBuilder.

THE POTION'S CATALYST

The toolsuite consists of an ETS kernel, Linktoc (a 32-bit linker/locator), shells for crossdebugging, and full support for C and C++ run-time libraries. When I opened this package of goodies, I thought I'd died and gone to heaven. The documentation is good. There are even null-modem cables!

The ETS kernel is a rudimentary OS that downloads to **EXPLR1**. If you can say "DOS" and know how to develop programs on larger systems with assembly, C, or C++, you can add embedded-systems program development to your bag of tricks.

The ETS kernel provides two important functions. It initializes 32-bit protected mode on the embedded target and provides facilities to run C and C++ run-time libraries. You can produce code as if it were targeted for a desktop PC.

The ETS kernel also communicates with a DOS-based PC. You can tap the inherent power of DOS programming tools and use your favorite Windows tools, too! ETS supports many Windows- and DOS-based compilers. I've chosen Visual C++ V. 4.0.

For EXPLRI, the ETS monitor loads into flash memory using Cyber Quest's Flash

listing 1: Control of the ***Intel '386EX*** power-management function is simply a matter of twiddling bits in the PWRCON register. Note that Normal Mode can be defined two ways.

```
#include "80386EX.h"
void Init_ClockPMU(void)
{
  _EnableExtIOMemO: /* Enable expanded I/O space for periph init.*/
/* Choose one of the three modes for your application */
  _SetEXRegByte(PWRCON, 0x0); /* Normal Mode */
  _SetEXRegWord(CLKPRS, ΩxΩ);
  _SetEXRegByte(PWRCON, 0x1); /* Normal Mode */
  _SetEXRegWord(CLKPRS, 0x1);
  _SetEXRegByte(PWRCON, 0x1); /* Powerdown Mode */
  _SetEXRegWord(CLKPRS, 0x0);
  _SetEXRegByte(PWRCON, 0x2); /* Idle Mode */
  _SetEXRegWord(CLKPRS, 0x0);
  _DisableExtIOMemO;          /* Restore I/O space.*/
}
```

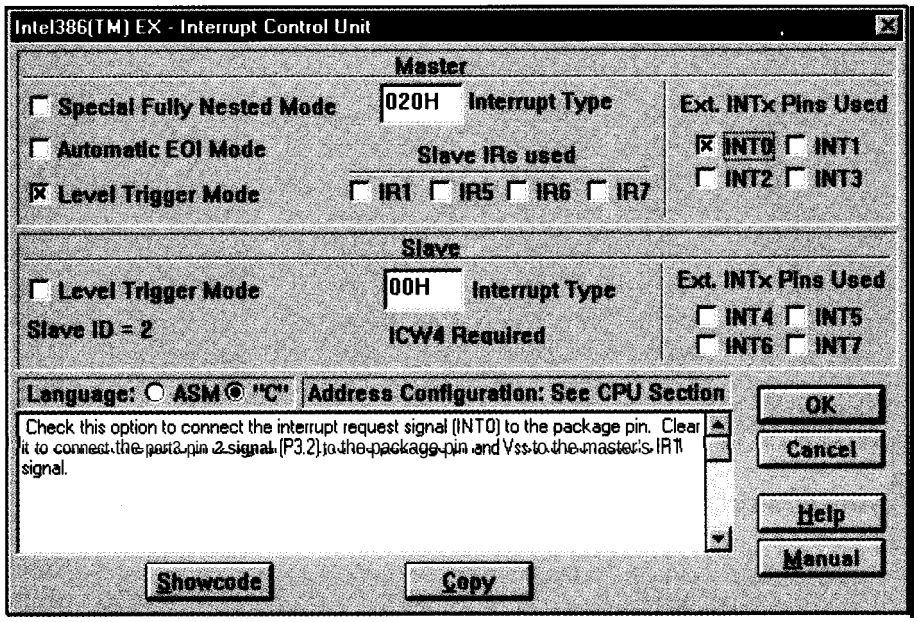


Photo 2: ***This sure beats*** thumbing through the Programmer's Reference Manual! Note that context-sensitive help ***is*** standard ***for these*** frames. In our example, a hint is given in reference to INTO.

File Loader. Once the monitor is loaded and kicked off on the embedded target, the host system communicates with the embedded monitor via a serial channel.

After you compile and link your embedded application, it downloads via the communications port to the target. Compilation is business as usual, but because I'm targeting an embedded platform with no real OS, linking is **done with** ETS Linktoc.

Linktoc is more than an ordinary program. It joins object modules from a broad base of assemblers and compilers for 'x86 embedded systems. It also provides full symbolic debugging support.

Glance at Photo 1. From this screen, you can reference the '386EX's manuals, configure peripherals and registers, inspect and verify instruction syntax, edit C or assembler code, and dive in for help.

Let's click the ApBuilder Peripher a l button and see who hollers.

AT THE STROKE OF MIDNIGHT

Like most micros, an external clock must spark the '386EX to life. It's accomplished

Listing 2: Excuse me for interrupting, ***but*** unlike many programmers, this application adds comments too!

```
/* All PC/AT-compatible internal peripherals should be in
   slot 0 of I/O space for DOS compatability.
Initialize the Interrupt Control Unit for:
   Slave 8259A and Master 8259A is mapped in slot 0.
Triggering: Master is level triggered: Slave is edge triggered.
Interrupt type Base: Master is 020H; Slave is OOH.
1 slave connected to IR2.
Fully nested mode: Master is Disabled: Slave is not allowed.
Automatic EOI mode: Master is Disabled; Slave is not allowed.
External interrupt pins used: INTO */

#include "80386EX.h"
void Init_ICU(void)
{
  _EnableExtIOMem(); /* Enable expanded I/O space for periph init.*/
  _SetEXRegByte(ICW1SDOS, 0x11); /* Set slave triggering */
  _SetEXRegByte(ICW2SDOS, 0x0);   /* Set slave base int type */
  _SetEXRegByte(ICW3SDOS, 0x2);   /* Set slave cascade pins */
  _SetEXRegByte(ICW4SDOS, 0x1);   /* Set slave IDs */
  _SetEXRegByte(ICW1MDOS, 0x19); /* Set master triggering */
  _SetEXRegByte(ICW2MDOS, 0x20);  /* Set master base int type */
  _SetEXRegByte(ICW3MDOS, 0x4);   /* Set master cascade pins */
  _SetEXRegByte(ICW4MDOS, 0x1);   /* Set slave IDs in master */
  _SetEXRegByte(INTCFG, 0x0);     /* Set external interrupt pins */
  _SetEXRegByte(P3CFG, 0x4);      /* Set external interrupt pins */
  _DisableExtIOMemO;              /* Restore I/O space.*/
}
```

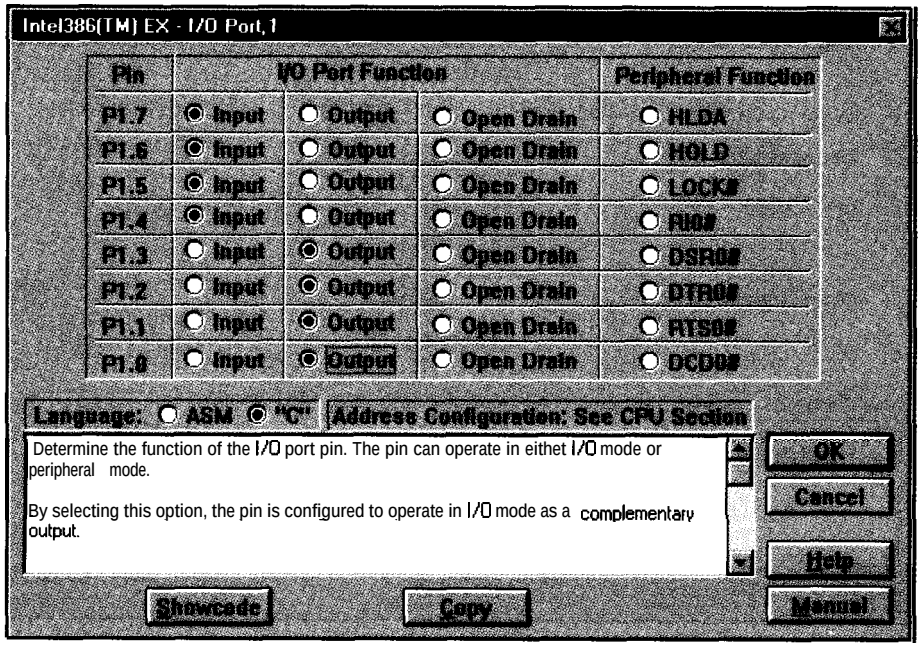


Photo 3: I thought port configuration wos easy on the good old 8255 parallel interface IC, and I had to do that by hand and manual! look at this! Reminds me of a Beatles tune, "I wanna hold your *hand."*

via the CLK2 input which provides the fundamental timing for the processor.

The clock- and power-management unit includes two divide-by-two counters and a programmable clock divider. The first counter divides the CLK2 frequency to generate a 50% duty-cycle clock signal. Independent clock signals are then routed to the core and the internal peripherals to implementthe power-management features.

A second counter divides the input fre quency again to generate SERCLK for the baud-rate generators of the asynchronous and synchronous serial I/O units. The SERCLK frequency is half the internal clock frequency (CLK2/4).

Another divider generates a prescaled clock (PSCLK) input for the timer/counter and synchronous serial I/O units. The minimum PSCLK frequency is the internal clock frequency divided by 2 (CLK2/4) and the maximum is the internal clock frequency divided by 5 13 (CLK2/1026).

The SIO (asynchronous serial I/O), SSIO (synchronous serial I/O), and timer/ countersareequipped withselectableclock

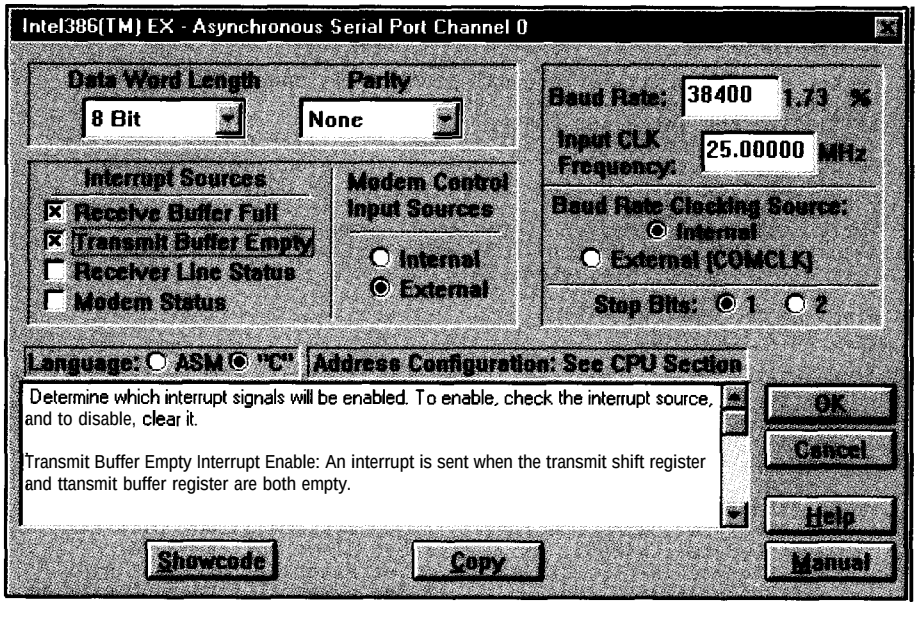


Photo 4: This *frame even shows the percentage of baud-rate error versus clock speed.*

listing 3: Although our Intel ***board comes up*** in Enhanced DOS Mode, note ***that ApBuilder*** provides the function *_EnableExtIOMem()* that enables expanded ***I/O*** space.

```
#include <stdio.h>
#include <stdlib.h>
#include "80386EX.h" /* This comes with ApBuilder.*/

/* Initialize I/O Port 1:
 PIN 0 = Output, PIN 1 = Output, PIN 2 = Output, PIN 3 = Output
 PIN 4 = Input, PIN 5 = Input, PIN 6 = Input, PIN 7 = Input */

void Init_IO1(void)
{
  _SetEXRegByte(P1LTC, 0xFF);
  _SetEXRegByte(P1DIR, 0xF0);
  _SetEXRegByte(P1CFG, 0x0);
}
void main()
{
  int i:
  int j;
  _EnableExtIOMem(); /* Enable expanded I/O for periph init.*/
    Init_IO1();
    for  (i = 0 ;i< 10  ;i++){
      j=_inpw(P1PIN);
    }
    printf("Exiting . . .\n");
    exit(0);
```

sources. The SIO unit uses either the SERCLK signal or an external clock connected to the COMCLK pin as its clock source.

The SSIO unit uses the SERCLK or PSCLK signal. The timer/counters use either the PSCLK signal or an external clock connected to the TMRCLK input pin.

RESET is always an asynchronous signal that requires special care. Within the '386EX, the asynchronous signal from the RESET pin is routed to the clock-generation unit, which synchronizes the **processor** clock with the falling edge of the RESET signal.

This process generates a synchronous internal RESET signal to the rest of the device. Internal logic ensures the correct clock phasing is initiated no matter when RESET occurs. Listing 1 tells us all how to manipulate the Power Control Register.

INTERRUPTS BREAK THE SPELL

The next peripheral is the Interrupt Control Unit (ICU). The ICU is composed of a pair of standard 82C59As configured as master and slave.

Each 82C59A can process eight interrupt-request (IR) signals. The master has seven interrupt sources and the slave 82C59A connected to its IR signals. The slave has nine interrupt sources connected

listing 4: Modern ***witches use modern technology like modems and serial*** ports. ***How else do you think they swap those delicious newt recipes?***

```
/* Initialize the Asynchronous Serial Port #0 for:
  Serial port 0 is mapped in slot 0.
  Word length of 8 bits, no parity, 1 stop bit
  Internal clocking: 25.00000-MHz clocking frequency
  External modem control sources: 38400(1.73%) bps baud rate
  Interrupt sources:
    Reception Complete Interrupt
    Transmission Register Empty Interrupt */

#include "80386EX.h"
void Init_SIO0(void)

  _EnableExtIOMem0; /* Enable expanded I/O space for periph init.*/
  _SetEXRegByte(SIOCFG, 0x1);   /* Set clocking and modem control.*/
  _SetEXRegByte(LCR0DOS, 0x80); /* Access divisor latch.*/
  _SetEXRegByte(DLH0DOS, 0x0);  /* Set divisor for baud rate.*/
  _SetEXRegByte(DLL0DOS, 0xA);
  _SetEXRegByte(LCR0DOS, 0x3); /* Set parity, stops, word size.*/
  _SetEXRegByte(IER0DOS, 0x3);  /* Enable selected interrupts.*/
  _SetEXRegByte(IIR0DOS, 0x0); /* Clear all interrupts at start.*/
  _SetEXRegByte(P1CFG, 0x0);    /* Set PxCFG reg for signal paths.*/
  _SetEXRegByte(P2CFG, 0x0);
  _SetEXRegByte(P3CFG, 0x1);
  _DisableExtIOMem();           /* Restore I/O space.*/
}
```

to its IR signals including twosources multiplexed into one IR signal.

Interrupts can be globally or individually enabled or disabled. The master asynchronously processes multiple interrupt requests at most any time.

A programmable priority structure determines a way to process multiple interrupts. When the master receives an interrupt, if it is enabled and has sufficient priority, the master sends the request via INT to the CPU.

This scheme lets interrupt requests be processed while another interrupt is being served. Thus, interrupts with higher priority can "interrupt" lower priority routines.

A slave 82C59A is cascaded from the master's IR2 signal and operates like the master. When the slave receives and decodes a valid interrupt request, it is sent to the master who routes it to the CPU.

Assume **we** configure the ICU as follows:

- Level Trigger Mode IRO Vector = 0x20
- INTO connects to an external package pin
- Slave is a don't care here

Clicking on I CU, we get Photo 2. Fill in the blanks and click Showcode. Listing2 is the resulting ApBuilder-generated code.

HOWLING WATCHWOLVES

Directly under ICU sits my watchdog, Spot. The '386EX watchdog timer consists of a 32-bit reload register, a 32-bit downcounter, an 8-state binary counter, and count and status registers.

Like all watchdog timers, the '386EX WDT recovers from unexpected program hangs. (Of course, I never use a WDT. Ha!)

The '386EX WDT takes the watchdog art a little further. In addition to traditional WDT duties, this puppy can be a general-purpose timer or operate in bus-monitor mode. Or, you can disable it entirely.

What's this bus-monitor mode? It protects normally **not-ready systems** from ready-hang conditions. When a bus cycle continues until the accessed peripheral device asserts *READY, the bus monitor times out.

This condition usually occurs when a bus cycle attempts to access an external peripheral in a nonexistent location.

THE CONJURE'S ALMOST DONE

The '386EX has three functionally identical 8-bit bidirectional I/O ports, each having threecontrol and one status register. Like comparable micros, the three I/O ports share pins with internal peripherals. If your design doesn't require a pin's peripheral function, you can individually configure that pin as an I/O port.

Each pin operates in I/O or peripheral mode. I/O mode controls the direction and value of each output pin. Peripheral mode gives this power to the peripheral.

Again, the '386EX takes the concept one step further. Most micros offer only two I/O pin configurations. Using I/O mode, a pin has three configurations: high-impedance input, opendrain output (with an external pullup), and complementary output.

Consulting the EXPLR 1 schematic, I found I/O Port 1 was available. Photo 3 shows the results of selecting I/O Port 1 and defining half the pins as output and half as input. The resulting code is in Listing 3.

THE RIGHT MUMBO JUMBO

Communications is my favorite subsystem. Its appeal is that there's only one way to do it. The '386EX is no exception.

Any asynchronous serial I/O subsystem has a baud-rate generator, transmitter, receiver, and some modemcontrol circuitry. Within the '386EX, the baud-rate generator can be clocked by the internal serial clock (SERCLK) signal or the COMCLK pin.

The transmitter and receiver consist of shift registers and buffers. Data to be transmitted is written to the transmit buffer and shifted out the transmit-data pin.

Conversely, received data is shifted in via the receive-data pin and transferred to the receive buffer. If you're on a modem, the modem-control handshaking signals come into play. Otherwise, you simply fake them out for a three-wire serial connection.

The trick is knowing how to set up the communications registers. Usually, you pour through bit maps and register layouts to get the bit group necessary. With ApBuilder, you click. Photo 4 and Listing 4 say it all.

BROOM 536, RUNWAY 5E CLEAR

I'd love to take you through a full development pass with this setup, but I can't single-step code in a picture.

I've introduced you to a mix of innovative software that, when combined with 'x86 hardware, is explosive. If you decide to implement a similar development system, check the vendor's tech support. Individual products work reliably, but some don't cooperate as a group.

I'll talk more about the Phar Lap and ApBuilder tools later, and remember-it doesn't have to be complicated to be embedded. APC.EPC

My special thanks to Marty Bakal and Jim Phillips at Phar Lap tech support for taking the broomstick when my nose was down!

Fred Eady has over 19 years experience as a systems engineer. He has worked with computers and communications systems large and small, simple and complex. His forte is embedded-systems design and communications. Fred may be reached at edtp@ddi. digital. net.

REFERENCES

Intel, *Intel* **'386EX Embedded** *Microprocessor User's* **Manual, Santa Clara, CA, V.** 272485-002, **1996.**

Intel, *Intel EXPLR 1* **Embedded** *PC Evaluation* **Platform** *Board Manual,* **Santa Clara, CA, V. 272775001, 1995.**

Phar Lap Software, *ETS* **Kernel User's Guide, Cambridge, MA, V.** TNTETSKern-1-1 **195, 1995.**

Phar Lap Software, Linkloc *Reference* **Manual, Cambridae. MA. 3rd Edition, 1995.**

Phar Lap Software, *CodeView Reference Manual,* **Cambridge, MA, 1 st Edition, 1993.**

Phar Lap Software, Using *the* **Phar Visual System Builder, Cambridge, MA, 2nd Edition, 1995.**

SOURCES

'386EX, ApBuilder
Intel Corp.
5000 W. Chandler Blvd.
Chandler, AZ 85226.3699
(602) 554-8080
Fax: (602) **554.7436**
BBS: **(9 16) 356-3600**
http://www.intel.com/

TNT Embedded ToolSuite
Phar Lap Software
60 Aberdeen Ave.
Cambridge, MA 02 138
1617) 661-1510
Fax: (617) 876-2972
BBS: (617) 661.1009
Telnet: bbs.pharlap.com

EXPLR1
RadiSys **Corp.**
15025 SW Koll **Pkwy.**
Beaverton, OR 97006-6056
(503) 646-I 800
Fax: (503) 646-I 850
BBS: (503) 646-8290
http://www.radisys.com/

PhoenixPICO
Phoenix Technologies
2770 De La Cruz
Santa Clara, CA 95050
(408) 6569000
Fax: (408) 452-I 985
http://www.ptltd.com/pico/

IRS

434 Very Useful
435 Moderately Useful
436 Not Useful

to its IR signals including twosources multiplexed into one IR signal.

Interrupts can be globally or individually enabled or disabled. The master asynchronously processes multiple interrupt requests at most any time.

A programmable priority structure determines a way to process multiple interrupts. When the master receives an interrupt, if it is enabled and has sufficient priority, the master sends the request via INT to the CPU.

This scheme lets interrupt requests be processed while another interrupt is being served. Thus, interrupts with higher priority can "interrupt" lower priority routines.

A slave 82C59A is cascaded from the master's IR2 signal and operates like the master. When the slave receives and decodes a valid interrupt request, it is sent to the master who routes it to the CPU.

Assume we configure the ICU as follows:

- Level Trigger Mode IRO Vector = 0x20
- INTO connects to an external package pin
- Slave is a don't care here

Clicking on ICU, we get Photo 2. Fill in the blanksandclick Showcode. Listing 2 is the resulting ApBuilder-generated code.

HOWLING WATCHWOLVES

Directly under ICU sits my watchdog, Spot. The '386EX watchdog timer consists of a 32-bit reload register, a 32-bit downcounter, an 8-state binary counter, and count and status registers.

Like all watchdog timers, the '386EX WDT recovers from unexpected program hangs. (Of course, I never use a WDT. Ha!)

The '386EX WDT takes the watchdog art a little further. In addition to traditional WDT duties, this puppy can be a general-purpose timer or operate in bus-monitor mode. Or, you can disable it entirely.

What's this bus-monitor mode? It protects normally not-ready systems from ready-hang conditions. When a bus cycle continues until the accessed peripheral device asserts *READY, the bus monitor times out.

This condition usually occurs when a bus cycle attempts to access an external peripheral in a nonexistent location.

THE CONJURE'S ALMOST DONE

'386EX has three functionally identical 8-bit

individually con-

'386EXie

pullup),

EXPLR1 schematic,

I/O Port

the results of selecting I/O Port 1 and defining half the pins as output and half as input. The resulting code is in Listing 3.

THE RIGHT MUMBO JUMBO

Communications is my favorite subsystem. its appeal is that there's only one way to do it. The '386EX is no exception.

Any **asynchronous** serial I/O subsystem has a baud-rate generator, transmitter, receiver, and some modemcontrol circuitry. Within the '386EX, the baud-rate generator can be clocked by the internal serial clock (SERCLK) signal or the COMCLK pin.

The transmitter and receiver consist of shift registers and buffers. Data to be transmitted is written to the transmit buffer and shifted out the transmit-data pin.

Conversely, received data is shifted in via the receive-data pin and transferred to the receive buffer. If you're on a modem, the modem-control handshaking signals come into play. Otherwise, you simply fake them out for a three-wire serial connection.

The trick is knowing how to set up the communications registers. Usually, you pour through bit maps and register layouts to get the bit group necessary. With ApBuilder, you click. Photo 4 and Listing 4 say it all.

BROOM 536, RUNWAY 5E CLEAR

I'd love to take you through a full development pass with this setup, but I can't single-step code in a picture.

I've introduced you to a mix of innovative software that, when combined with 'x86 hardware, is explosive. If you decide to implement a similar development system, check the vendor's tech support. Individual products work reliably, but some don't cooperate as a group.

I'll talk more about the Phar Lap and ApBuilder tools later, and remember-it

APC.EPC

My special thanks to Marty Bakal and Jim Phillips at Phar lap tech support for taking the broomstick when my nose was down!

Fred Eady has over I9 years experience as a systems engineer. He has w computers and communications systems large and small, simple and complex. s fort embedded-systems design and com-

edtp@ddi.digital.net.

REFERENCES

Intel, *Intel '386EX Embedded Microprocessor User's Manual,* **Santa Clara, CA, V. 272485.002, 1996.**

Intel, *Intel EXPLR 1 Embedded PC Evaluation Platform Board Manual,* **Santa Clara, CA, V. 272775001, 1995.**

Phar Lap Software, *ETS* **Kernel User's** *Guide,* **Cambridge, MA, V.** TNTETSKern-1-1 **195, 1995.**

Phar tap Software, *LinkLoc Reference Manual,* **Cambridge, MA, 3rd Edition, 1995.**

Phar tap Software, *CodeView Reference Manual,* **Cambridge, MA, 1 st Edition, 1993.**

Phar top Software, Using the *Phar Lop Visual System Builder,* **Cambridge, MA, 2nd Edition, 1995.**

SOURCES

'386EX, ApBuilder
Intel Corp.
5000 W. **Chandler Blvd.**
Chandler, AZ 85226-3699
(602) 554.8080
Fax: (602) 554-7436
BBS: (916) 356.3600
http://www.intel.com/

TNT Embedded ToolSuite
Phar Lao Software
60 Aberdeen Ave.
Cambridge, MA 02 138
,16171, 661-1510
Fax: {617} 876-2972
BBS: {617) 661-1009
Telnet: bbs.pharlap.com

EXPLR1
RadiSys **Corp.**
15025 SW Koll **Pkwy.**
Beaverton, OR 97006-6056
(503) 646-I 800
Fax: (503) 646-I 850
BBS: (503) 646-8290
http://www.radisys.com/

PhoenixPICO
Phoenix Technologies
2770 De La Cruz
Santa Clara, CA 95050
(408) 654-9000
Fax: (408) 452-I 985
http://www.ptltd.com/pico/

IRS

434 Very **Useful**
435 Moderately Useful
436 Not Useful

CONNECTIME

conducted by Ken Davidson

The Circuit Cellar BBS
300/1200/2400/9600/14.4k bps
24 hours/7 days a week
(860) 871-I 988-Four incoming lines
Internet E-mail: **sysop@circellar.com**

Last month, I promised some special offerings for those readers who can't frequent the BBS as often as they'd like. I'm going to try something new for a few months to see how you like it.

There are frequently discussions on the BBS that go on far longer than I'd ever have space to print here. In the past, I've tried to extract portions of such long discussions to use for the column. However, many times there is so much good information that I'd be doing the discussion, BBS users, and readers a disservice by trying to extract a small piece.

In addition to printing a selection of short threads, I'm going to select a handful of long discussion threads each month and summarize them here. I'll give you the exact phrase used in the subject field of the messages on the BBS. You'll then be able to search the message base if you want to call the BBS directly and quickly download the thread to read offline.

I'm also going to place the full text of those message threads on the Software on Disk for the month so you can get them if you don't have a modem or if a phone call costs too much.

Some of you may remember when I offered BBS on Disk. I'd capture an entire month's BBS message traffic and place it on three disks. Preparing those became too time consuming to continue, but I think you'll enjoy this "best of BBS" offering in its place.

For each of the following summaries, the subject header is the exact text found in the "Subject:" field of the messages I'm describing. If you call the BBS to capture the messages, simply tell the BBS to search by subject field and enter the phrase to search for.

Ground scheme

The first thread I've selected started with a rather innocent question from someone asking why, when he connects the ground of his scope to the circuit he's measuring, something always blows up. An engineer he was working with told him to defeat the ground prong on the scope's power cord. He wants to know why.

Well, I've described in the past what a religious war is, and I've printed examples of some mild wars. However, this thread ended up exploding into one major war.

Some of our very experienced and respected users took the side that under no circumstances should the ground protecting ever be defeated and that there are appropriate methods for making such measurements safely.

Others fought back with the argument that, if you know what you're doing and are very careful, such measurements can be done safely and, indeed, are necessary in some cases.

While there is no "correct" side to be on, each opponent presents some fascinating arguments and case histories supporting their side of the issue. The thread is well worth reading. Be sure to be your own judge, though.

What is "quality"?

This is another thread where case histories and personal experiences make for some interesting reading. Someone poses the question of just what is quality. Does it describe a product's appearance in addition to its functionality, or does it simply reflect its reliability?

Someone points out that while we in the U.S. often downplay the technological significance of countries such as Russia, we still need multibillion-dollar stealth bombers to avoid their Mig fighters and air defenses.

Is newer technology necessarily better when the old, no matter how kludgy, still does the job?

Help with circuit?

The next thread starts with someone asking for help in designing a circuit that can measure currents from 10 nA to 10 mA, giving an output voltage proportional to the current measured. The input impedance may also range from 1 kΩ to 1 TΩ [that's *teraohm!).* One better, the voltage used to develop the current is 10 or 100 VDC and is user selectable.

Our callers bat around quite a few design ideas and issues, making for a very interesting exchange.

DTMF trans from PIC

Next, we look at what it would take to generate DTMF tones using nothing but a PIC processor. The main reason for wanting to use the PIC instead of a dedicated chip is the potential for high volumes and the desire to keep the parts count low.

After some initial discouraging notes, the original poster directs our attention to a Microchip application note de-

scribing how to solve the problem on a PIC 17C42. However, for this situation, the '17C42 costs too much and uses too much power.

The discussion continues, and the necessary external component count continues to rise to the point that a dedicated chip ends up being cheaper.

There are some neat tidbits about DTMF specs, filtering, and sampling.

DTMF decode/encode

On a related topic, another caller asks how hard it would be to do DTMF decoding using a DSP or a 20-MHz PIC. And since we're doing DTMF decoding, how about encoding and call-progress monitoring?

The general consensus is that a PIC might be able to do some of the operations, and a DSP likely could handle it. A few users even threw around the idea of using a Motorola 'HC05 to do the chore.

There are some interesting facts discussed related to DTMF signaling and DSP architectures.

IC testing

Msg#: 5113
From: Eric Mielcarek To: Ed Nisley

Since you've been helpful beyond words, I thought I might bounce another thing off you. I was discussing a problem concerning a product we manufacture at work.

The Problem: the device activates itself at room temperature
The Suspect: an optocoupler
Function: the opto controls a FET to turn on the device

My coworker suspects that an increase in temperature causes enough leakage through the triac output of the opto to bias the base of the FET. I tend to agree.

However, his method for proving this out is, to me, questionable. He holds a heat gun on the opto (it is a narrow-point gun) until it activates the FET-average time about 10 s. In that 10 s, the temperature reaches approximately 150°C. The maximum spec is 70°C.

Although IC testing is usually done at temperatures close to or at tolerance, I feel this not a valid test. What do you think? Thanks in advance for your input.

Msg#: 5502
From: Ken Simmons To: Eric Mielcarek

Pardon my interrupt, but have you checked the bias level of the opto's input LED? It's possible the LED is biased real close to its emission (turn-on) point, which would then unwittingly activate the opto's triac output when the temperature was "just right" for sufficient leakage.

Perhaps a series resistor or maybe a series diode (or two?) with the input driver to raise the turn-on point a little bit?

Just my two bytes on this.

Msg#: 6376
From: Ed Nisley To: Eric Mielcarek

Uh oh. Self-activating devices. Run and hide!

Smelting the board with a heat gun certainly flushes out any weak parts, but cooking it until the traces just about drip off seems like overkill to me. After all, a lot of *other* parts get pushed out of their normal operating specs, so you really haven't shown very much.

A better approach would be to clip out a little aluminum square that fits the top of the optocoupler, dial your soldering iron as low as it'll go, and carefully heat the poor thing. If it fails with a little gentle heat (not a bake!), that'11 be more convincing. Don't cook it, though! Perhaps a little thermocouple will keep the temperature within reason.

Even better, look at the circuit and figure out *why* the FET is so sensitive. There ought to be a pull-down or other bias circuit in addition to that switched input. Otherwise, you've got a FET with a floating gate and that's known to be a Bad Thing.

Add a pull-down resistor that's low enough to sink the expected leakage without turning the FET on, verify that it works correctly, then install a bigger resistor that fails consistently. That'11 give you an idea of the range of your problem.

If you've really got a floating gate, the right fix is an engineering change rather than a tighter spec on the coupler!

Msg#: 6907
From: Eric Mielcarek To: Ed Nisley

Once again Ed, thanks for your input. I also agreed that a heat gun could flush out the weak parts. Knowing this, I also believe it should not be used as a method of finding production-based problems. With all the new test technology currently available, it is a mystery why someone would use a heat gun instead. That method should be saved until there are no other alternatives. I just wanted to make sure I wasn't the only one that feels that way. Thanks again.

Msg#: 7736
From: Ken Simmons To: Eric Mielcarek

What's the problem? I use temperature chambers operating at 165°F daily to flush out weak parts in completed assemblies. Also use -65°F in the same chamber for the same thing.

CONNECTIME

Msg#:11441
From: Eric Mielcarek To: Ken Simmons

The only problem I can see, Ken, is at that temperature you are exceeding the manufacturer's specifications. In my opinion, at that temperature, weak parts are being created. It only takes a second or two for that temperature to reach the die and possibly cause damage.

Msg#:13449
From: Ken Simmons To: Eric Mielcarek

True, however "a second or two" should not cause any damage, permanent or otherwise. Sustained operation at that temperature, that's another story altogether.

If you're fortunate enough to use mil-spec-rated parts, with temperature ratings of -75 to 175°F, then temperature really isn't a factor.

Msg#:16845
From: Eric Mielcarek To: Ken Simmons

Agreed. But we are not using mil-spec parts. Also, the test was done in excess of 10 s with a minimum temperature of 160°C. This kind of exposure is a little much for the component being tested.

Msg#:18193
From: Ken Simmons To: Eric Mielcarek

I see. I do remember the databook warnings of "No more than 300" on any pin for more than 5 s" in regards to soldering.

And 160°C is pretty high!

IR sensor floodlights

Msg#:21307
From: Ralph Willing To: All Users

Outside, under the eaves of my house, I have a floodlight that is triggered by an IR sensor. It has a photocell to keep it from coming on during the day. A few weeks ago, it started coming on during the day with or without motion in the area.

It's located above the gable vent, so I'm wondering if the hot air from the vent could be affecting it.

Has anyone taken one of these puppies apart? Are there any user-adjustable or -replaceable parts inside?

Given that it's 20' up in the air, I might take a stab at it, but only if I know there's a chance of success.

Otherwise, it's probably easier to just replace the whole unit since they're only about 15 bucks at Home Depot.

Msg#:22084
From: Steve Ciarcia To: Ralph Willing

Get a ladder and see if something has just built a nest over your photosensor. ;-)

Msg#:23638
From: Ralph Willing To: Steve Ciarcia

Been there, done that.

Didn't bother to take it apart while I was up there. Think I'll just replace it.

I'm like George (see Msg#:23 105) in that I'll probably just put it in a box in my basement after I take it apart to see what makes it tick.

Msg#:23781
From: Ken Simmons To: Ralph Willing

Sounds like the controlling triac has shorted out. I'd check that first before digging into the IR circuitry.

Msg#:23105
From: George Novacek To: Ralph Willing

> Has anyone taken one of these puppies apart? Are there
> any user-adjustable or -replaceable parts inside?

Yes, many times. There are some adjustments you can make, but this is probably not the problem. It's more likely that some of the components are dying.

I have been using these "puppies" for years. Most of them don't last more than 18 months in our Canadian climate. I have a box of the dead ones in a box in my basement. I'm too cheap to throw them out and am kidding myself that some days I will have nothing better to do than fix them. In view of their cost, they're hardly worth fixing.

As far as the circuit goes, they're simple. One or two pyroelectric sensors, a photoresistor, a three-terminal regulator, a rectifier, a quad op-amp, and a small relay with a transistor driver is all you need.

The most damage-sensitive part is the pyroelectric sensor, but since your unit flashes even during the daylight, the problem can be anywhere.

Msg#:23639
From: Ralph Willing To: George Novacek

Like you, I'll probably start a box in my basement, unless you want to add this one to your collection?

Thanks for the info. I'll probably take this one apart to see what makes it tick.

Msg#:24116
From: George Novacek To: Ralph Willing

I can give you a very fast rundown of how the PIR switch works.

CONNECTIME

You start with one or two pyroelectric sensors. All the commercial switches use Fresnel lens arrays to give it a certain field of view. Generally, a mirror optical system is more versatile and has less insertion loss, but it is more expensive to build, generally bulkier, and Racal has a patent which keeps everybody else away from it.

Realistically, you can't get much better than a 90" field from a sensor, so many switches now include two sensors, which gives them a minimum 180" vision. The sensor output when detecting a human body within about 100' range is in the millivolts.

So now you need an amplifier followed by a comparator so the millivolt signal is brought up to a nice switching level of a comparator. You need about 60 dB (x1000) gain in the amplifier.

You need to frequency control the amp's characteristic. It starts about 0.005 Hz, growing at 20 dB/decade up to about 5 Hz (full gain of 1000), where it starts dropping off at a rate of about 40 dB/decade.

The following comparator has a one-shot timer and an AND gate tied to a photoresistor to lock out the switch during daylight. The comparator needs to be a window comparator since the PIR output will be bipolar, depending on the direction of the detected movement.

You have three controls, all of them usually built into the comparator circuit. Because you need the amplifier to act as a fairly stable band-pass filter, controlling the gain is not very easy.

It is simpler to modify the size of the comparator window to modify detection sensitivity. The second control modifies the one-shot time constant to control the duration of the light on. Finally, the third control is tied to the photoresistor to set daylight-detection level.

The comparator drives a lamp switch. A triac could be used, but I have only seen relays used. This may be because the PIR is very sensitive to RF emissions, and triac switching-even with a zero-crossing driver-might be a bit unpredictable.

Mechanically, the most important aspect is to prevent air from moving across the PIR sensor(s) and maintain the focus-the lenses are l-2" focal length. The air movement could create minute temperature changes on the surface of the sensor, which leads to false triggering.

Don't forget, with right optics the PIR sensor can detect a surface-temperature change of a fraction of a degree several hundred feet away!

I had one switch which false triggered. Once I put a tape across the openings for the controls, the problem disappeared.

Internally, the sensors are completely wrapped in a plastic sleeve to prevent their metal body from being heated up by electronic parts. An LED in the vicinity will do it.

Msg#:24160
From: Ralph Willing To: George Novacek

Thanks for the tutorial. I'll probably get the ladder out again this weekend and take it down to tear it apart.

Before I posted the first message, I went up there to see what effect the exterior adjustments would have, but nothing helped. That was when I noticed the strong hot-air current coming out of the louver, so you may be correct about that. Just strange that it's worked for six years and now decides to act up.

We invite you to call the Circuit Cellar BBS and exchange messages and files with other Circuit Cellar readers. It is available 24 hours a day and may be reached at (860) 871-1988. Set your modem for 8 data bits, 1 stop bit, no parity, and 300, 1200, 2400, 9600, or 14.4k bps.

ARTICLE SOFTWARE

Software for the articles in this and past issues of *Circuit Cellar INK* may be downloaded from the Circuit Cellar BBS free of charge. It is also available on the Internet at http://www.circellar.com/. For those with just E-mail access, send a message to info@circellar.com to find out how to request files through E-mail.

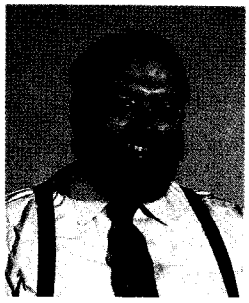
Message threads summarized at the beginning of the column are also available on the BBS for at least six months after they are first posted. The subject line at the start of each summary matches the subject used on the messages themselves. Simply call the BBS and search for those subject lines to find the message threads.

For those unable to download files or messages, the software and messages are also available on disk. Software for issues prior to 1995 comes on a 360-KB IBM PC-format disk, one issue per disk. For issues from 1995 on, software comes on a 1.44-MB PC-format disk, with three issues per disk. Disks cost $12 each. To order Software on Disk, send check or money order to: Circuit Cellar INK, Software On Disk, P.O. Box 772, Vernon, CT 06066, or use your Visa or Mastercard and call (860) 8752199. Be sure to specify the issue numbers with your order. Please add $3 for shipping outside the U.S.

IRS

437 Very Useful 438 Moderately Useful 439 Not Useful

The Radical Fringe



Usually, it's about 20 minutes before shipping the magazine to the printer as I hurriedly rush into the editorial office with a copy of my latest ramblings. Janice gives me the usual "about time" nod and mumbles a few things under her breath about still not having it in a form she can use. By the time I can get it to her in digital format, I may as well have sent it over by smoke signals.

This time around, I sat down a whole day in advance just to see if I could beat the rat race. As you might expect, I drew a complete blank!

After about an hour of typing, backspacing, correcting, and erasing the same first line, I decided to look and see what topics were hot among other editorial types. I sat back with a bunch of trade journals and even a few financial publications (now that they think technology isn't a dirty word). Certainly among this persuasive group, I'd get enough of an advance scoop on the new "10-GHz '986 processor," "wireless virtual LAN," or latest "500-MHz laptop with pullout wet bar" to vent all my frustrations about everything being just a lot of vaporware anyway.

Much to my amazement, a lot of the people writing editorials these days are jovial. It's also nice to see that there are even a few others who, like me, might be considered on the deep fringe when it come to dissertation topics.

One particularly eloquent techie is Bob Pease. Like me, he seems to have been around forever. Indirectly, he was responsible for successes in my early engineering years. If he had not joined National Semiconductor in time to make all their second-source linear devices from Fairchild Semiconductor actually work, a bunch of us engineers might now be accountants instead. Bob made the LM741 and LM108 do for analog back then what all the gate arrays, ASICs, and programmable logic are doing for digital today.

One specific Bob Pease feature has convinced me that my anxiety about relevant editorial is groundless-people on the fringe can get away with anything. The commentary that woke me up, published in *Electronic* Design, was about how Bob counts and categorizes "dead cars" along the highway. You can even SASE him for his list going back to 1969 if you share a similar persuasion.

When Bob passes a disabled car along the road, he jots down the manufacturer and other pertinent details. Of course, if a guy with a Volvo is on the side of the road with a VW Rabbit and both have their hoods up, he tends to count that as ½ Volvo, ½ Rabbit, and 1 helper. Helpers are not counted as dead cars if their purpose is obvious. Similar lists are kept for how many people are changing tires or pulled over by the police-even down to the number of cars with broken drive shafts (in 1990, there was a high of 16).

Perhaps living in California makes all this seem so humorous to me yet deadly serious to Bob. Out there, Bob might get away driving his 300,000-mile-plus '68 VW Beetle waving a sign saying, "You Have No Brake Lights." On this end of the map, you have to worry about drivers from Massachusetts who generally don't know what signal or brake lights are, and there are always those guys who want to reciprocate "with extreme prejudice" when they see someone shaking something provocatively in their direction.

Understand that this isn't criticism. The difference between Bob and myself may only be the level of technology. While I don't keep lists, I have often thought of using a side-window scrolling LED display to communicate with other drivers. Until then, I'll just wave a note scratched on the back of an envelope, too.

By the way, Bob, the next time you are driving through Connecticut, if you come across a blue BMW, I'd appreciate some assistance, not just a dead-car listing.

Steve

steve.ciarcia@circellar.com