

# CIRCUIT CELLAR

## INK<sup>®</sup>

THE COMPUTER APPLICATIONS JOURNAL

#55 FEBRUARY 1995

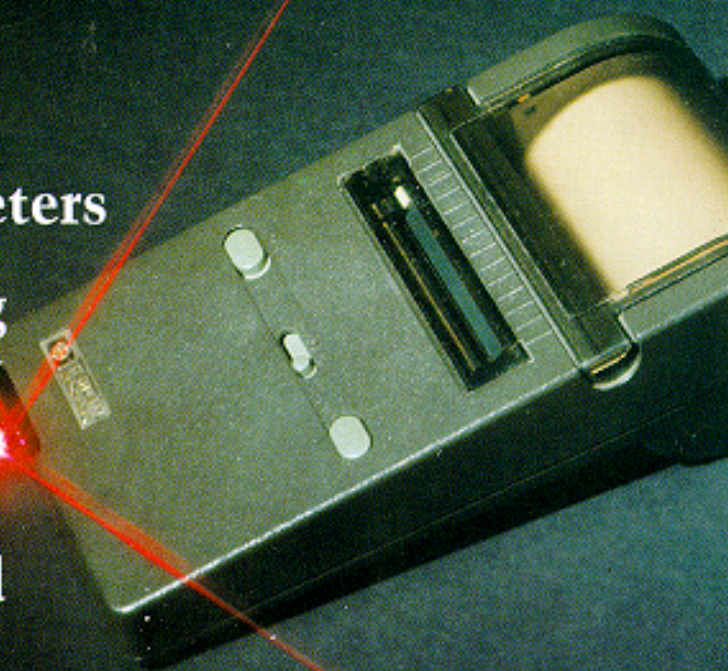
## EMBEDDED INTERFACING

Interfacing to Flow Meters

Infrared-based Printing

Battery-operated  
Power Supplies

AT89C251 Flash-based  
Microcontroller



\$3.95 U.S.  
\$4.95 Canada



# EDITOR'S INK

## Jelly Beans



Although conventional wisdom holds that embedded applications require special embedded controllers and interfaces, we're starting to see that wisdom challenged. When cost, not size, is the primary concern, off-the-shelf solutions from the desktop arena often offer a better price/performance ratio than their semicustom cousins.

A good example is Ed's embedded '386SX series. He's been able to show that by using inexpensive, "jelly-bean" motherboards instead of tiny, specialized controllers, costs stay down without sacrificing performance.

I had occasion in the past few weeks to come across another example, though with a much smaller market. Caller ID, a service provided by the local phone companies across the U.S., provides the called party with the caller's phone number (and sometimes name) before answering the phone. Caller ID boxes that display and record the data are available from many electronics suppliers in a price range of \$30-\$90. Most of these units are stand-alone devices, though. Interfacing them with a computer to make further use of the information is a costly and kludgy proposition.

Some of these boxes do have a serial port tacked on the side, but it drives the price up well over \$100 for the unit. There is a better alternative.

If you can do without the fancy display (e.g., your application uses a voice to notify you who's calling), many modems on the market today offer Caller ID as an additional feature to the normal data and fax capabilities. Why pay over \$100 for a box that does nothing but Caller ID when you can get a modem to do the whole deal for under \$130?

Another good example of reusing other solutions can be found in one of our articles this month. Infrared-based printers intended for use with intelligent calculators are inexpensive and readily available. Using them in an embedded application is a simple matter of knowing what IR codes to send. Why pay triple the price for a special-function, battery-operated printer when a low-cost, canned solution exists?

Such economies of scale helped put a virtual end to build-your-own PCs (though no amount of monetary savings could match the pleasure many of our readers derive from making their own). This trend is now moving into the embedded peripheral market where A/D converter boards, Caller ID boxes, and tiny printers have so many features included that it doesn't make sense to roll your own. The functional is already there! And who has the time or money, anyway?

There will always be a place for custom and semicustom solutions, especially with constraints on size, weight, power usage, and so forth. Such engineering is our lifeblood. We just have to be sure to keep an open mind when design specs aren't tight.

# CIRCUIT CELLAR®

THE COMPUTER APPLICATIONS JOURNAL

FOUNDER/EDITORIAL DIRECTOR  
Steve Ciarcia

PUBLISHER  
Daniel Rodrigues

EDITOR-IN-CHIEF  
Ken Davidson

PUBLISHER'S ASSISTANT  
Sue Hodge

TECHNICAL EDITOR  
Janice Marinelli

CIRCULATION COORDINATOR  
Rose Mansella

ENGINEERING STAFF  
Jeff Bachiochi & Ed Nisley

CIRCULATION ASSISTANT  
Barbara Maleski

WEST COAST EDITOR  
Tom Cantrell

CIRCULATION CONSULTANT  
Gregory Spitzfaden

CONTRIBUTING EDITOR  
John Dybowski

BUSINESS MANAGER  
Jeannette Walters

NEW PRODUCTS EDITOR  
Harv Weiner

ADVERTISING COORDINATOR  
Dan Gorsky

ART DIRECTOR  
Lisa Ferry

GRAPHIC ARTIST  
Joseph Quinlan

PRODUCTION STAFF  
John Gorsky  
James Soussounis

CONTRIBUTORS:  
Jon Elson  
Tim McDonough  
Frank Kuechmann  
Pellervo Kaskinen

CIRCUIT CELLAR INK, THE COMPUTER APPLICATIONS JOURNAL (ISSN0896-8985) is published monthly by Circuit Cellar Incorporated, 4 Park Street, Suite 20, Vernon, CT 06066 (203) 675-2751. Second class postage 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 696, 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

## HAJAR ASSOCIATES NATIONAL ADVERTISING REPRESENTATIVES

NORTHEAST & MID-ATLANTIC  
Barbara Best  
(908) 741-7744  
Fax: (908) 741-6823

SOUTHEAST  
Christa Collins  
(305) 966-3939  
Fax: (305) 985-8457

WEST COAST  
Barbara Jones & Shelley Rainey  
(714) 540-3554  
Fax: (714) 540-7103

MIDWEST  
Nanette Traetow  
(708) 789-3080  
Fax: (708) 789-3082

Circuit Cellar BBS—24 Hrs. 300112001240019600114 4k bps.6 bits, no parity, 1 stop bit, (203) 871-1988; 24001 9600 bps Courier HST, (203) 871-0549

All programs and schematics in *Circuit Cellar INK* have been carefully reviewed to ensure their performance is in accordance with the specifications described, and programs are posted on the Circuit Cellar BBS for electronic transfer by subscribers.

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

Entire contents copyright © 1995 by Circuit Cellar Incorporated. All rights reserved. Reproduction of this publication in whole or in part without written consent from Circuit Cellar Inc. is prohibited.

**1 4** **Interfacing Flow Meters to High-speed Counters**  
*by Bill Payne*

**1 8** **Use Infrared to Make Embedded Printing Easy**  
*by Jeff Fisher*

**2 2** **It's Not Just for Memory Anymore**  
An Introduction to PCMCIA  
*by Lalo J. Gastriani*

**3 0** **Speeding and Slimming Your Port Access**  
A Different Way of Reading from the PC Parallel Port  
*by Tracey Lee & Kok-Leong Ong*

**3 6** **Battery-operated Power Supplies**  
Selecting the Right Battery and Supply for Your Application  
*by David Prutchi*

**5 0**  **Firmware Furnace**  
Journey to the Protected Land: Infrastructure Improvement  
*Ed Nisley*

**6 0**  **From the Bench**  
Fitting 10 oz. into a 5-oz. Package  
An Application for Highway Safety  
*Jeff Bachiochi*

**6 8**  **Silicon Update**  
I Sync, Therefore I DRAM  
*Tom Cantrell*

**7 6**  **Embedded Techniques**  
Downsizing  
Atmel's AT89C2051 Flash-based Microcontroller  
*John Dybowski*

# INSIDE ISSUE 55

**2** **Editor's INK**  
Ken Davidson  
**Jelly Beans**

**6** **Reader's INK**  
Letters to the Editor

**8** **New Product News**  
edited by Harv Weiner

**82** **ConnecTime**  
Excerpts from  
the Circuit Cellar BBS  
conducted by  
Ken Davidson

**96** **Steve's Own INK**  
Steve Ciarcia  
**Once Every  
27,000 Years**  
**Advertiser's Index** **81**

# READER'S INK

## NEW COVER-A STEP UP

I got the latest *INK* today-very professional looking. It reminds me of *Popular Science* or such. Of course, the contents are *much better* than those other magazines!

Hope y'all have a great '95!

Russ Reiss  
Bolton, CT

## PRAISE AND FIXES

I really enjoy Ed Nisley's series on protected-mode programming. At the moment, my only application is writing assembly language subprograms to run under Lahey FORTRAN F77L EM/32, which runs under the Pharlap DOS extender. Even so, I've found several helpful nuggets of information. If Ed wants to explain how to write an interrupt service routine for IRQ2 (vertical retrace) running in protected mode under Pharlap, I wouldn't mind at all.

As an ancient controls engineer (MIT '57), I noticed that Tom Cantrell's PID-pong control loop has the gains set wrong. The response shown in Figure 5 (*INK* 50) is much too oscillatory. You can see, not quite hidden by the noise, a slowly decaying sine wave in the fan plot starting after each change in the position setpoint.

The frequency of the sine wave is the natural frequency of the entire system; it shouldn't be so prominent in the responses. In a well-designed system, it would damp out in a couple of cycles or less after a change in setpoint. With this design, the motor has to work too hard and there is little stability margin. A few decibels gain increase (temperature or whatever) could make the design unstable.

To improve the response, set the gain to zero. Try reducing the *P* gain or increasing the *D* gain. Up to a point, increasing the *D* gain allows for higher values of the *P* gain without making the response too oscillatory. Once you get good values for the *P* and *D* gains, increase the *I* gain as far as possible without spoiling the transient response. Small test inputs should be used in this stage. Finally, adjust the nonlinear parameters for the largest inputs you expect.

James C. Wilcox  
Palos Verdes Estates CA

P.S.: Thanks for sending a replacement for the issue that the postal service mangled.

*Editor's Note: The March issue features fuzzy logic and will be offering two fuzzy-pong designs. Be sure to pick up a copy so you can compare Tom's technique with a fuzzy approach.*

## FOR YOUR INFORMATION

Thank you for featuring *fuzzyTECH-MP*, Microchip Technology Inc.'s new fuzzy logic development tool, in your December '94 issue. Several readers have asked us how to obtain more information about Inform Software Corporation, Evanston, Ill., who developed the fuzzy logic tool suites for this device. Inform Software can be reached at (800) 929-28 15. The *fuzzyTECH-MP* product is available only through any Microchip worldwide sales office and authorized dealer or by calling (602) 786-7668.

Eric Sells  
Public Relations Manager  
Microchip Technology, Inc.

## Contacting Circuit Cellar

We at Circuit Cellar *INK* encourage communication between our readers and our staff, so 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 (203) 875-2199.

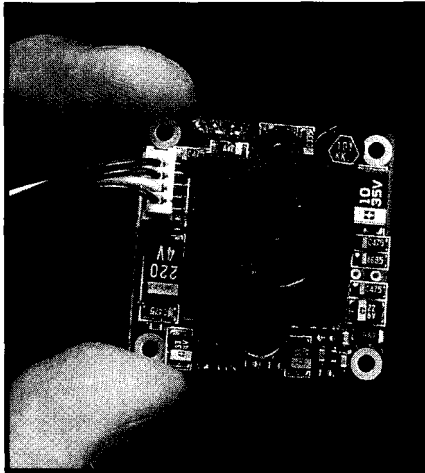
**Fax:** All faxes may be sent to (203) 872-2204.

**BBS:** All of our editors and regular authors frequent the Circuit Cellar BBS and are available to answer questions. Call (203) 871-1988 with your modem (300-14.4k bps, 8N1).

**Internet:** Electronic mail may also be sent to our editors and regular authors via the Internet. To determine a particular person's Internet address, use their name as it appears in the masthead or by-line, insert a period between their first and last names, and append "@circellar.com" to the end. For example, to send Internet E-mail to Jeff Bachiochi, address it to [jeff.bachiochi@circellar.com](mailto:jeff.bachiochi@circellar.com). For more information, send E-mail to [info@circellar.com](mailto:info@circellar.com).

# NEW PRODUCT NEWS

Edited by Harv Weiner



## MINIATURE CCD CAMERA

A miniature breadboard-type CCD black & white camera (1.26" x 1.26" x 0.75") is available from Edmund Scientific. The camera can be used for home and office security, robotics, machine vision, and instrumentation applications.

The unit produces video output with an automatic electronic shutter and features a 3.7-mm pinhole lens built into the breadboard assembly. The field of view is 45" vertical x 60" horizontal with 32-mm x 32-mm dimensions. Miniature details can be observed on a TV monitor from a range of only inches up to 15' with a 0.3-lux minimum illumination. Resolution is 380 TV lines horizontal by 350 lines vertical. The unit uses 9-VDC input.

Edmund Scientific Co.

Dept. 14B1, N999 Edscorp Bldg. • Barrington, NJ 08007  
(609) 573-6259 • Fax: (609) 573-6295

#500

## EMBEDDED SYSTEM DEVELOPMENT TOOLS

Phar Lap's new TNT **Embedded ToolSuite** delivers a total solution for 32-bit embedded-systems development on the Intel 386/486 family of microprocessors. The suite offers an easy and cost-effective, one-stop solution for building embedded applications using popular DOS and Windows compilers. Supported compilers include 32-bit C/C++ products from Borland, Microsoft, and MetaWare.

The ToolSuite's components include the TNT Embedded Kernel, Visual System Builder, LinkLoc (a 32-bit linker and locator), and CVEMB and TDEMB shells for embedded cross-debugging. Full support for C/C++ run-time libraries, an MS-DOS-compatible file system, and a floating-point-emulation library are also included.

The TNT Embedded Kernel provides a simple operating system for running embedded programs. Its two main functions are to initialize 32-bit protected mode and provide the foundation for running a C/C++ run-time library. These functions can save development time since developers don't have to set up their own protected-mode environment or learn new run-time libraries. The Kernel can be loaded from ROM or diskette. The Kernel also includes a floating-point emulator, native MS-DOS file system, and a remote file system.

The Visual System Builder enables developers to configure and customize the TNT Kernel to match their hardware setup through an easy-to-use Windows utility. Developers can specify where memory is located in the target, how the kernel is loaded into memory, and how the target system communicates with the debugger.

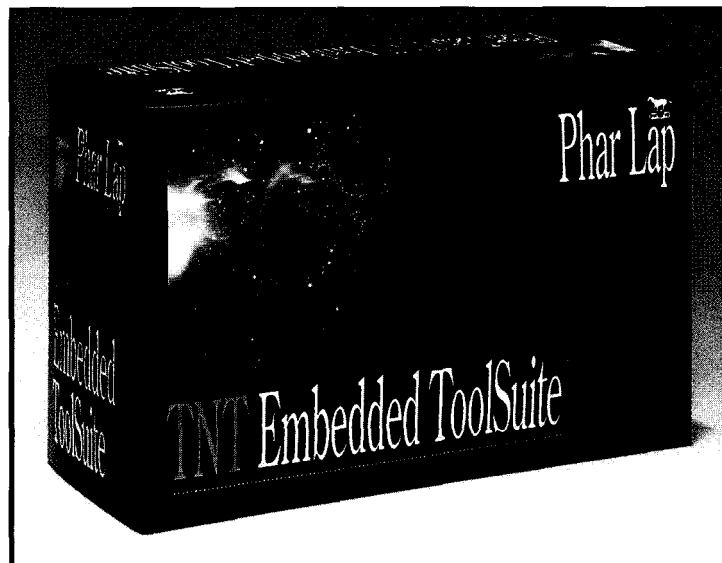
As a sophisticated one-step linker/locator, LinkLoc builds embedded applications that run on the TNT Embedded Kernel. LinkLoc includes a rich set of command switches to control the entire link process. It can provide full symbolic information for C/C++ source-level debugging with CodeView or Turbo Debugger. In addition, it can output files for an in-circuit emulator or PROM programmer.

The TNT Embedded ToolSuite sells for \$2995. Requirements include a PC-compatible host computer running DOS and Windows 3.1.

Phar Lap Software, Inc.

60 Aberdeen Ave. • Cambridge, MA 02138  
(617) 661-1510 • Fax: (617) 876-2972

#501



# NEW PRODUCT NEWS

## HIGH-SPEED ANALOG I/O MODULE

Real Time Devices introduces the **DM5408 dataModule**, an ultracompact, PC/104-compliant data acquisition and control module for OEM embedded and portable applications. The DM5408 features channel-gain scan memory, 1024-sample A/D buffer, and bit-programmable digital I/O with two advanced digital-interrupt modules.

The DM5408 features 12-bit A/D converter at 200 kHz with 8 differential or 16 single-ended channels. Inputs are jumper configurable for three ranges: -5 to +5, -10 to +10, or 0 to 10 V. Programmable gains of 1, 2, 4, and 8 can be combined with the input ranges to optimize gain/resolution tradeoff. A 1024-sample buffer provides an interface between the A/D converter and PC to enhance triggering and storage capabilities and to maintain data integrity. Four high-speed, digital-input lines, synchronized to the conversion rate and stored in the buffer, can be used as a 4-bit data or trigger marker without interrupting conversion.

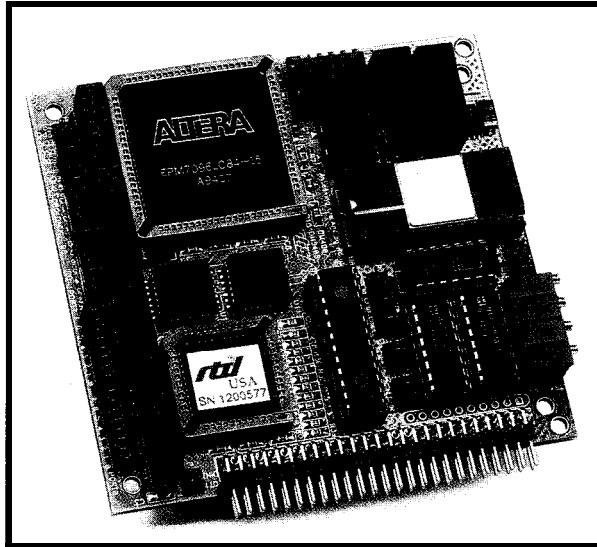
The channel-gain scan memory offers channel sampling in any order with a different gain on each channel at high speeds. A skip bit is included to support different sampling rates on different channels, thereby saving memory and eliminating the need to discard unwanted data. Digital output lines are also provided to control TMX32 analog-input expansion boards for high-speed scanning of up to 5 12 random channels. The DM5408 has several triggering and conversion modes supported through the hardware and software including pre-, post-, and about-trigger, channel scanning, and random- and multiburst.

The bit-programmable digital I/O lines support two advanced digital-interrupt modes. An interrupt can be generated when the lines match a programmed value or when any bit changes its current state. There are also two 12-bit DACs and two 16-bit timer counters.

The DM5408 sells for \$450 in 100-piece quantities.

Real Time Devices, Inc.  
P.O. Box 906 • 200 Innovation Blvd.  
State College, PA 16804 • (814) 234-8087 • Fax: (814) 234-5218

#502



Capacities range from 170 to 810 MB and access time is 12 ms. The unit weighs only 11 oz. and is less than 1" x 3" x 6" in size. Pluggers are EPP compliant and pass printer information through untouched. The Plugger ships as a complete kit, which includes a carrying case, DOS and OS/2 drivers, keyboard and mouse power adapters, AC wall plug, and installation guide. No controller card is needed. Pluggers support Stacker and Double Space which increases capacity to well over 1 GB.

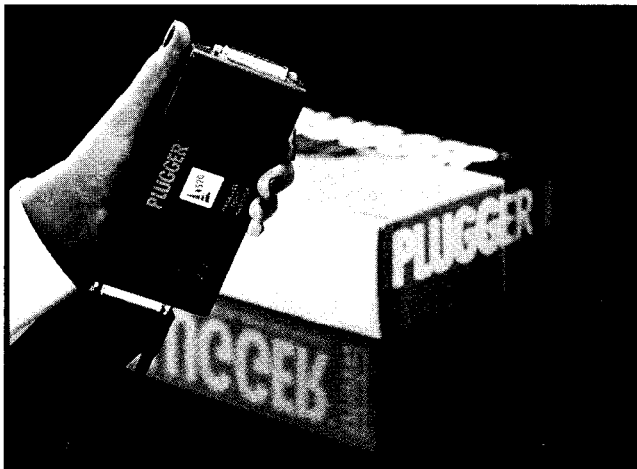
Plugger list prices range from \$499 to \$999 and have a two-year warranty.

Computer Connections  
America  
19A Crosby Dr.  
Bedford, MA 01730  
(617) 271-0444  
Fax: (617) 271-0873

#503

## PARALLEL PORT HARD DRIVE

Computer Connections America has introduced **Plugger**, a 2.5" hard drive that installs on any parallel printer port. Plugger does not require additional hardware for DOS and OS/2 systems and no CONFIG.SYS changes are needed. Plugger is an ideal solution for mobile storage expansion, file transfer, backups, and PC installations.



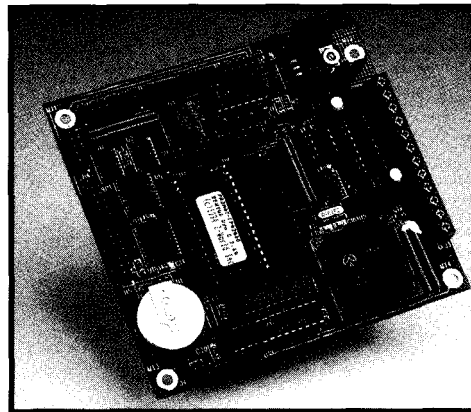
# NEW PRODUCT NEWS

## C-PROGRAMMABLE CONTROLLER

The Little Genius from Z-World is a compact, low-cost controller appropriate for control and data-acquisition applications. Standard features include a battery backup for RAM and real-time clock, watchdog timer, power-failure interrupt, EEPROM for system constants, up to 5 12 KB RAM and up to 512 KB EPROM (32 KB standard), and two Z180 DMA channels.

The Little Genius uses a 9.216-MHz Z180 microcontroller and is packaged on a 4.5" x 4.2" board. Fully populated, the Little Genius provides 12 digital inputs, 14 digital outputs, serial communication, programmable timers, back-up battery, and large memory space. A connector to the PLCBus provides room for expansion boards; a piggyback prototyping board is being developed.

The unit has been designed to accommodate keypads or arrays of contacts of up to 84 elements, relays,



solenoid actuators, solid-state relays, and many kinds of LCDs. There are two asynchronous ports that can be configured as two RS-232 channels or one RS-232 and one RS-485 channel. The ports operate up to 57.6 kbps.

The Little Genius is supported by Z-World's Dynamic C development system which combines a text editor, fast compiler, and symbolic debugger. Dynamic C comes with dozens of function libraries (including a real-time

kernel] and sample programs in source code. It is available for DOS and Windows.

The Little Genius sells for \$149 and includes a schematic, manual, and wall transformer. Other configurations and options are available.

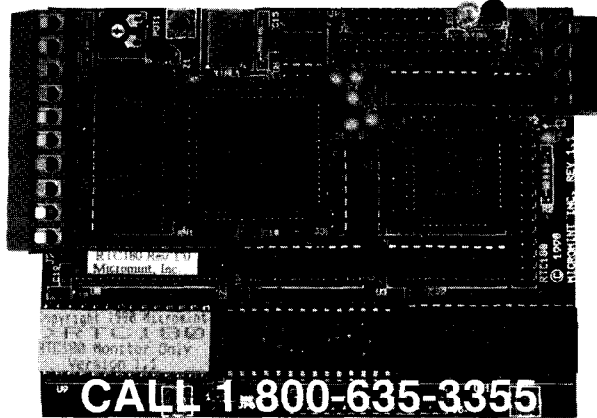
Z-World Engineering

1724 Picasso Ave. • Davis, CA 95616  
(916) 757-3737 • Fax: (916) 753-5141

#504

# STOP LOOK LISTEN

Odds are that some time during the day you will stop for a traffic signal, look at a message display or listen to a recorded announcement controlled by a Micromint RTC180. We've shipped thousands of RTC180s to OEMs. Check out why they chose the RTC180 by calling us for a data sheet and price list now.



**CALL 1-800-635-3355**

**MICROMINT, INC.**

4 Park Street, Vernon, CT 06066  
(203) 871-6170 • Fax (203) 872-2204



in Europe: (44) 0285-658122 • in Canada: (514) 336-9426 • in Australia: (3) 467-7194 • Distributor Inquiries Welcome

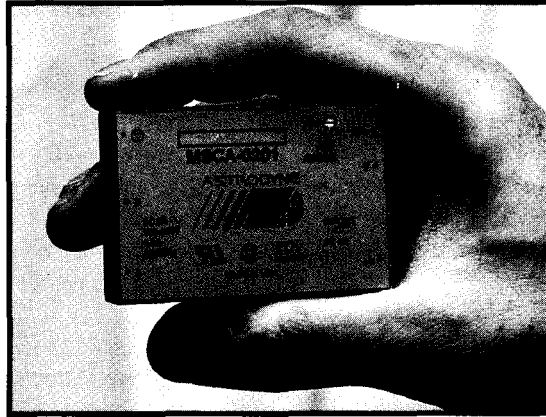
#119

# NEW PRODUCT NEWS

## ULTRAMINIATURE SWITCHING POWER SUPPLY

Designed to meet the compact switching-power-supply requirements of portable systems, embedded controls, and medical equipment, the M-Series AC/DC switching power-supply modules from Astrodyne accept universal input over the ranges of 85-265 VAC and 110-340 VDC as well as providing single, dual, and triple outputs in voltage ranges of 3.3-24 VDC.

Measuring just 2.55" x 1.77" x 0.83", these ultraminiature modules provide power densities of up to 2.8 W per cubic inch and include unique features such as an integral potentiometer for output-voltage adjustment and an LED power-status indicator. Remote power on and off control is provided via a single logic-level control pin, making the M-Series ideal for use in energy-saving designs in which auto shutdown of power is desirable.



The M-Series of AC/DC switching modules provide input-to-output isolation of 3750 VAC and are available in both commercial and low-leakage UL-544 medical versions. The modules meet UL, CSA, and TUV safety and FCC and VDE Level B EMI requirements. They also include output-overvoltage protection circuitry. The M-Series power supplies operate over the ambient temperature range of 0-50°C without the output derating necessary when using convection cooling. Models are available in a wide variety of packaging and interconnection styles.

M-Series power supply prices start at \$59 (single output).

Astrodyne  
412 High Plain St.  
Walpole, MA 02081  
(508) 668-2311  
Fax: (508) 668-9942

#505

Astrodyne  
412 High Plain St.  
Walpole, MA 02081  
(508) 668-2311  
Fax: (508) 668-9942

THE PARADIGM

OUTTA MY LIFE, VERMIN!



If, like Doc, you think bugs belong six feet under, then step up to Paradigm DEBUG and get the right weapon for the toughest '186 or V-series embedded application. Take 'em on by yourself with Paradigm **DEBUG/RT** or gang up on 'em with a popular in-circuit emulator. 'Cuz no matter what kind of Borland or Microsoft C/C++ vermin you're fightin', ya better not go in empty-handed or firing blanks from inferior weapons. Ya just might not live to regret it.

'Nuff said.

## PARADIGM

Proven Solutions for Embedded C/C++ Developers

**I-800-537-5043**

Paradigm Systems  
3301 Country Club Road, Suite 2214, Endwell, NY 13760  
(607) 748-5966 / FAX: (607) 748-5968  
Internet: 73047.3031@compuserve.com

TO BE CONTINUED...

©1994 Paradigm Systems, Inc. All rights reserved.



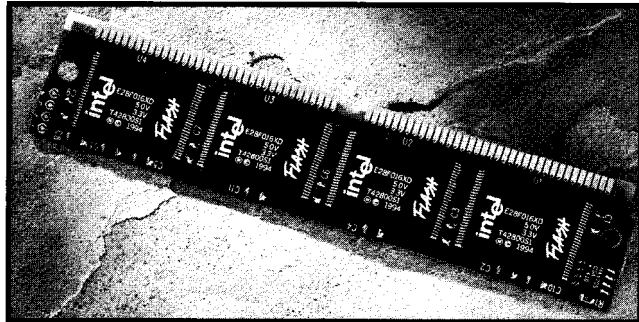
# NEW PRODUCT NEWS

## FLASH-MEMORY MODULES

Smart Modular Technologies has developed 4- and 8 MB SIMMs using Intel's new embedded flash-RAM integrated circuits which combine the high-speed readability of DRAMs with the nonvolatile update capabilities of flash. Designated as the **SM532F1000** (4 MB) and **SM532F2000** (8 MB), the new CMOS flash modules are targeted at high-performance embedded applications which previously stored programs or executable files in ROM or on disk and downloaded them into volatile DRAM for operation.

The flash-RAM modules are ideal for storing instantly executable programs such as BIOS, DOS and other operating systems, Windows application packages, fonts for printers, and other executable code. The modules thus save space on a system's hard drive while offering instant-on operation.

The 4-MB module is organized as 1 megaword by 32 bits, and the 8-MB module is organized as 2 megawords by 32 bits. Both modules are packaged in 72-pin leadless SIMMs. Control signals are also DRAM standard. The SIMMs incorporate a DRAM interface, which accommodates RAS/CAS signals and multiplexed address lines.



The SIMMs are therefore capable of working with existing DRAM controllers. Control lines permit 16-bit word control.

Because the flash-RAM chips are nonvolatile, no refresh cycles are needed to retain data, a factor that further lowers CPU overhead and saves power.

Pricing of the new flash-RAM SIMM is \$149.85 for 4-MB and \$279.65 for 8-MB (in quantity).

Smart Modular Technologies, Inc.  
45531 Northport Loop West • Fremont, CA 94538  
(510) 623-1231 • Fax: (510) 623-1434

#506

The  
only  
8051/52  
BASIC  
compiler  
that is  
100 %  
BASIC 52  
Compatible  
and  
has full  
floating  
point,  
integer,  
byte & bit  
variables.

- Memory mapped variables
- In-line assembly language option
- Compile time switch to select 8051/8031 or 8052/8032 CPUs
- Compatible with any RAM or ROM memory mapping
- Runs up to 50 times faster than the MCS BASIC-52 interpreter.
- Includes Binary Technology's SXA5.1 cross-assembler & hex file manip. util.
- Extensive documentation
- Tutorial included
- Runs on IBM-PC/XT or compatible
- Compatible with all 8051 variants
- **BXC51 \$ 295.**

**508-369-9556**  
**FAX 508-369-9549**



Binary Technology, Inc.  
P.O. Box 541 • Carlisle, MA 01741



## Cross Assemblers

- Local Labels and Cross Reference
- Powerful Macro Substitution Capability
- Machine Cycle Counting
- 32 Significant Character Labels and Symbols
- Unlimited Include File Capability
- Selectable Intel Hex or Motorola Hex Object File

## Simulators

- Source View Symbolic Debugging
- Attach Keyboard, Screen or Data Files to Simulate I/O
- Machine Cycle Counting
- Ten User-definable Screens
- Unlimited Breakpoints, Memory and I/O Mapping
- Trace File to Record Simulator Session
- Ability to Step Backward through Simulation

## Disassemblers

- Automatic Substitution of Defined Label Names for All Jumps and Branches

\*Automatic Insertion of Supplied Comments and Expressions

## Application Source Libraries

- 16 and 32 bit Integer Arithmetic and Numeric/String Conversion

## PseudoCorp

921 Country Club Road, Suite 200 • Eugene, OR 97401

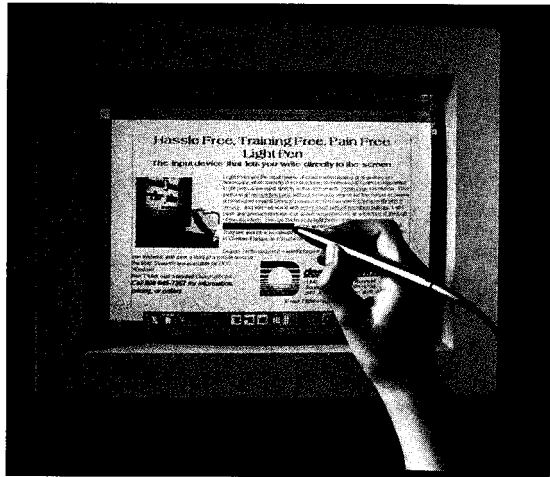
(503) 683-9173

Fax: (503) 683-9186 BBS (503) 683-9076

#105

#106

# NEW PRODUCT NEWS



## JITTER-FREE LIGHT-PEN INTERFACE

Design Technology has introduced a unique light-pen interface board. The all-digital DT360 offers fast, smooth operation. The light pen is available for DOS 3.3+, Windows 3.1+, or NeXTSTEP 3.2+.

The DT360 is a complete system on a board and features an onboard processor and clock crystal. With much less reference to the CPU, the DT360 is faster, more accurate, and has much smoother cursor moves. Because the jitter has been designed out, there is no lag normally associated with the smoothing process. In some modes, the smoothness is visibly superior to the mouse cursor.

The DT360 features locking-function options that prevent the installation of any other pen or board in the system. The DT360 can send and receive codes so that if an unauthorized replacement pen is plugged in, it simply won't work. Similarly, if

the proprietary software sends a code when an unauthorized interface board has been installed, it won't work either. This gives VARs and OEMs better control of what gets installed in their systems.

The DT360 interface accepts all Design Technology styles of light pen from the basic black anodized aluminum to a double side-switch, stainless-steel model.

Evaluation systems start at \$188 with a money-back guarantee.

Design Technology • 11489 Woodside Ave. • Santee, CA 92071-4724 • (619) 448-2888 • Fax: (619) 448-3044

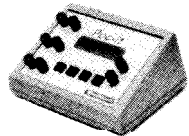
#507

**If you design:**  
Circuits, Systems, or  
Embedded Software

ONLY  
**\$295**

Then you need:

**Poc-it**



Power-On Cycler - Intermittent Tester

The circuits, boards, and systems we design today are complex. We are using more complex components to create these designs. Thus, we must test these components over a wide range of scenarios. For example, we have found VLSI chips, software libraries and hardware designs that **intermittently** fail to **powerup** properly. To minimize costly re-work and embarrassing failures, we must test these designs across a large number and wide variety of power-up scenarios. Will your design power up every time?

**Poc-it** is designed specifically to help you test your design for just these scenarios. Featuring:

- 10 amp 120 VAC receptacle
- 10 amp relay
- Two 5 VDC high speed inputs
- One IO-30VDC sense input
- Outputs easily programmable from 0.01 sec to 100 min.



714 Hopmeadow St., P.O. Box 624, Simsbury, CT 06070  
(800) 651-6170 FAX: (203) 651-0019

#107

# Smart

Little Genius™

# Controller

Starting at \$119, Qty 1

Our C-programmable miniature controllers are ideal as the brains for control applications, data acquisition, and test and measurement. Features include digital I/O to 400 lines, ADCS, DACS, relays, solenoid drivers, RS232/RS485, battery-backed RAM, clock, watchdog, LCDs, keypads, enclosures and more. Use our simple, yet powerful, Dynamic C™ development system (\$195 integrated editor, compiler and debugger) for quick project completion!

24-Hour AutoFax 916.753.0618. Call from **your** FAX. Request catalog 18.

1724 Picasso Ave. Davis, CA 95616 916.757.3737 916.753.5141 FAX

#108

## FEATURES

14

Interfacing Flow Meters to High-speed Counters

18

Use Infrared to Make Embedded Printing Easy

22

It's Not Just for Memory Anymore

30

Speeding and Slimming Your Port Access

36

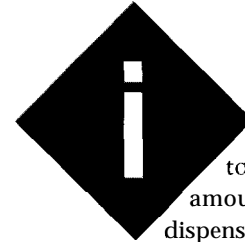
Battery-operated Power Supplies

# Interfacing Flow Meters to High-speed Counters

After introducing us to the basics of flow meters, Bill delves more deeply into how to get around a noisy analog situation. His special trick—debouncing the magnetic reed switch in a high-speed application.

## FEATURE ARTICLE

**Bill Payne**



Bill recently needed to measure the amount of fluid dispensed in a customer application. The project involved interfacing a high-speed digital counter on an embedded controller with a electromechanical flow meter, an application commonly found in beverage, medication, and fuel dispensing systems. You find some type of flow meter anywhere fluid flow must be measured. The problems, however, start when mixing the analog signals from the flow meters with high-speed digital systems.

Figure 1 depicts a basic flow meter. It consists of a paddle wheel attached to a cam which opens and closes a switch as it rotates. By placing the paddle wheel into the fluid flow, a pulse rate is produced which is proportional to the flow rate.

Electromechanical flow meters can be purchased in two basic models. The newer versions use a rotating magnet and a Hall-effect transducer. These are straightforward since the switch element is a semiconductor device. The more common type of flow meter uses a rotating magnet and a magnetic reed switch. Figure 2 depicts the waveforms at the opening and closing of the reed switch. As shown, the reed switches are bouncy on both opening and closing.

The majority of flow meter applications do not operate at a high rate of speed. As an example, the flow meters in a gasoline dispenser only provide ten pulses per gallon. Just think of how long (in microprocessor cycles) it takes to pump a gallon of gas into your vehicle's gas tank.

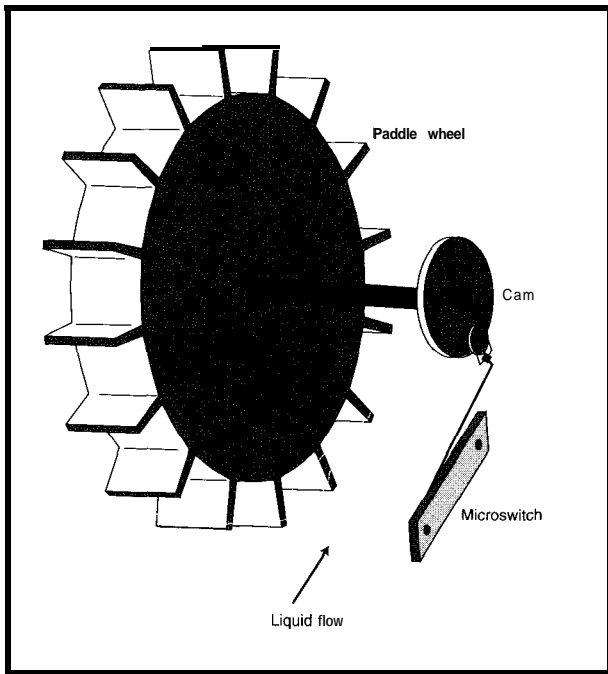


Figure 1—In a typical fluid-flow measuring system, a reed switch is closed on each revolution of the paddle wheel.

Unfortunately, most semiconductor systems do not care about such low-speed environments. Instead, they count every bounce of the magnetic reed switch as a pulse transition. When you realize that the reed switch may open and close more than one hundred times and your flow rate is only ten pulses per gallon, the problem becomes obvious.

Figure 3 depicts a common interface to a high-speed counter. The digital counter is a standard Intel 82C54. This device has a maximum counter-input frequency of 10 MHz. The counter is connected to the outside world through an optoisolator. In my opinion, this is a requirement when working with any control system. The optoisolator provides a separation between the noisy analog (real world) system and the digital system.

The opto also provides security for the digital system in the event of a catastrophic failure on the analog side. Some of the newer optoisolators have internal Schmitt triggers to increase their frequency response. This

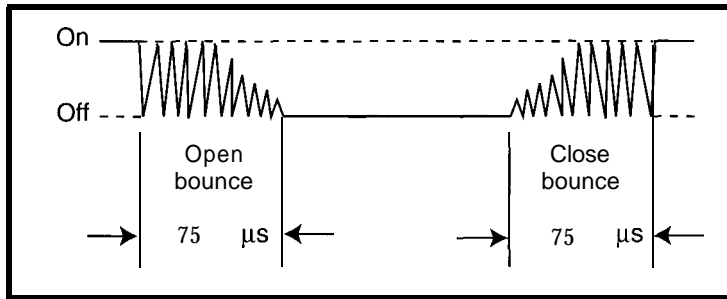


Figure 2—The jagged pattern marks the opening and closing contact bounce for a magnetic reed switch.

is fine for high-speed communications systems, but it is not a virtue when interfacing with slow-speed devices such as flow meters.

The design in Figure 3 has several limitations in an interface with real-world analog systems.

There are two different environments to consider when designing signal-conditioning circuits for high-speed counters: one which has fairly clean signal and power leads and the other which has AC waveforms capacitively coupled to the signal and power leads.

## DC ENVIRONMENT

The signal conditioning necessary in a DC environment is usually related to the bounce of the magnetic reed switch. One technique used quite effectively debounces the reed switch with a 555 timer IC.

This device can be set to virtually any time span through the selection of

one timing resistor and one capacitor. It can sink or source up to 200 mA of current from or to an output device, which is more than enough current to drive the LED in an optoisolator. It can operate over a 5-15-V range. And most importantly, the internal timing operation is fairly insensitive to power fluctuations. This enables the device to be used in environments that don't have much filtering on the supply.

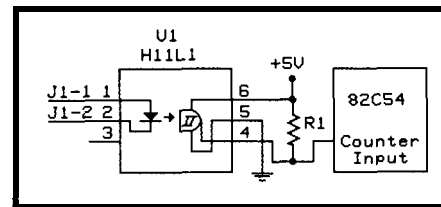


Figure 3-A simple optoisolator can be used to interface a flow meter to a counter in a relatively slow and quiet environment.

The circuit in Figure 4 can be used to clean up the output from the reed switch in the flow meter. This circuit uses a dual 555 timer IC referred to as a 556 IC. Both timer circuits in the 556 IC are configured the same. J1 pin 1 and J2 pin 1 are the outputs that drive

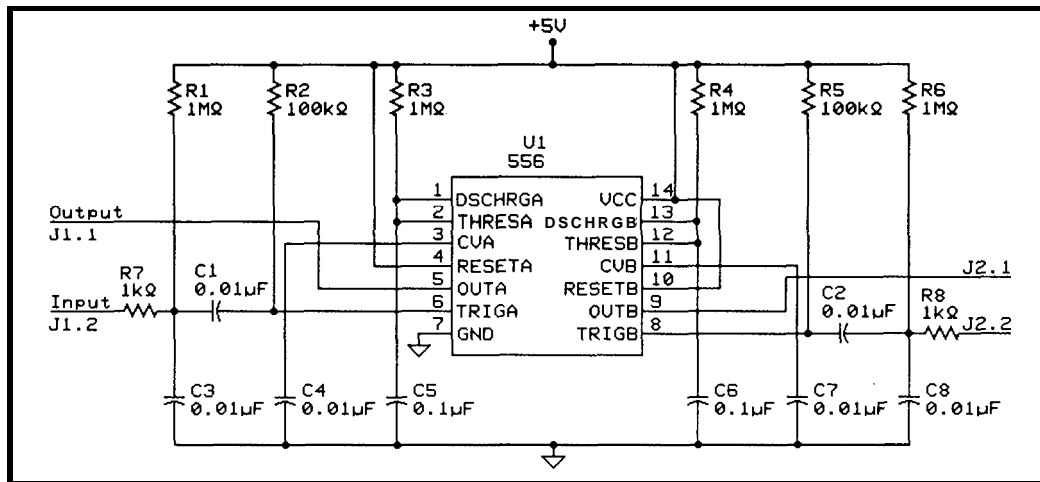


Figure 4-A 556 can be used to clean up the raw flow-meter signals and generate clean, square pulses.



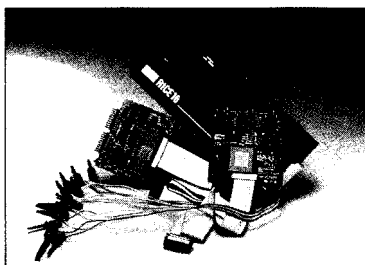
# PIC16C5x/16Cxx Real-time Emulators

Introducing RICE16 and RICExx-Juniors, real-time in-circuit emulators for the PIC16C5x and PIC16Cxx family microcontrollers: affordable, feature-filled development systems from **\$599\***

\*Suggested Retail for U.S. only

## RICE16 Features:

- Real-time Emulation to 20MHz for 16C5x and 10MHz for 16Cxx
- PC-Hosted via Parallel Port
- Support; all oscillator type5
- 8K Program Memory
- 8K by 24-bit real-time Trace Buffer
- Source Level Debugging
- Unlimited Breakpoints
- External Trigger Break with either "AND/OR" with Breakpoints
- Trigger Outputs on any Address Range
- 12 External Logic Probes
- User-Selectable Internal Clock from 40 frequencies or External Clock
- Single Step, Multiple Step, To Cursor, Step over Call, Return to Caller, etc.
- On-line Assembler for patch instruction
- Easy-to-use windowed software
- Support 16C71, 16C84 and 16C64 with Optional Probe Cards
- Comes Complete with TASM16 Macro Assembler, Emulation Software, Power Adapter, Parallel Adapter Cable and User's Guide
- 30-day Money Back Guarantee
- Made in the U.S.A.



Emulators for **16C71/84/64** available now!

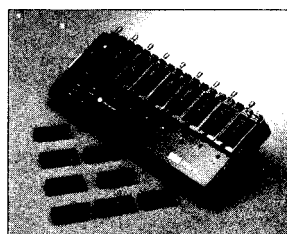
## RICE-xx Junior series

RICE-xx "Junior" series emulators support PIC16C5x family, PIC16C71, PIC16C84 or PIC16C64. They offer the same real-time features of RICE16 with the respective probe cards less real-time trace capture. Price starts at \$599.

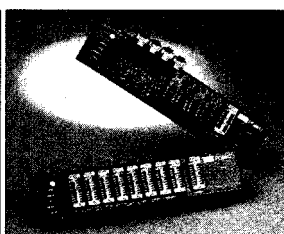
## PIC Gang Programmers

Advanced Transdata Corp. also offers PRODUCTION QUALITY gang programmers for the different PIC microcontrollers.

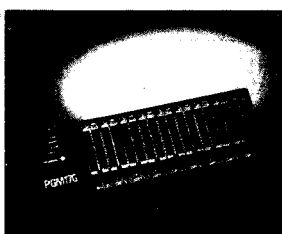
- Stand-alone COPY mode from a master device
- PC-hosted mode for single unit programming
- High throughput
- Checksum verification on master device
- Code protection
- Verify at 4.5V and 5.5V
- Each program cycle includes blank check, program and verify eight devices
- Prices start at **\$599**



PGM16G: for 16C5x family



PGM47: for 16C71/84



PGM17G: for 17C42

Call **(214) 980-2960** today for our new catalog.

For RICE16.ZIP and other product demos, call our BBS at (214) 980-0067.



**Advanced Transdata Corporation** Tel **(214) 980-2960**  
14330 Midway bad, Suite 120. Dallas, Texas 75244 Fax (214) 980-2937

the optoisolator LEDs. J1 pin 2 and J2 pin 2 are connected to the reed switches within the flow meters. This circuit can operate over a 5-15-V DC range. Using 12 V increases the noise immunity for the reed switch over long distances.

## CIRCUIT OPERATION

Prior to the closure of the reed switch, capacitor C3 is charged to 12 V through resistor R1. The closing of the reed switch rapidly discharges capacitor C3 through resistor R7. All bounce from the reed switch is removed by the integrating action of capacitor C3 and resistor R1.

This clean, negative-going spike is then fed into the trigger input of the 555 timer through capacitor C1. This trigger pulse fires the 555 timer generating an output pulse of

$$\text{Time} = 1.1 \times R3 \times C5.$$

This output pulse then drives the LED within the optoisolator. On release of the reed switch, capacitor C3 recharges to 12 V and the circuit waits for the next switch closure.

The time constants and trigger-input filtering send a single pulse for each closure of the reed switch. With this design, the circuit does not retrigger when the reed switch is held in the closed position. Instead, it outputs a single pulse and waits for the next trigger pulse. You can tune the debounce period by simply altering the values of the timing resistor and capacitor.

## AC ENVIRONMENT

The circuit in Figure 4 works fine as a debouncer for the magnetic reed switch in a low-noise environment. When put into an environment such as Figure 5, the circuit can fail miserably. The flow-meter signal lead becomes a low-impedance antenna when the magnetic reed switch is open. The induced AC waveform on the signal lead can be as much as 2530 VAC during this time. The waveform occurs because of the capacitive coupling between the conductors encased in the conduit. The integrator input section composed of R1 and C3 in Figure 4

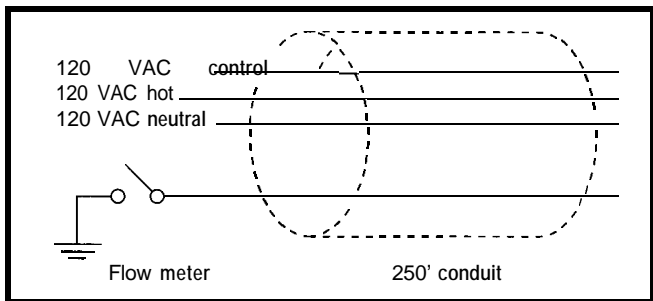


Figure 5-h industrial control applications, 60-cycle noise from AC lines that are often contained in the same conduit as the raw flow-meter output adds to the front-end filtering problem.

cannot handle this type of noise. This inability combined with the AC coupling of the 555 trigger input through C I means you are counting 60-cycle AC when the reed switch is open.

Figure 6 is a simple, yet effective, way to counter the induced noise from the other conductors in the conduit. You can calculate the exact values for the pull-up resistor and filter capacitor by mathematically modeling both the signal conductor and the magnetic reed switch. You can then solve their

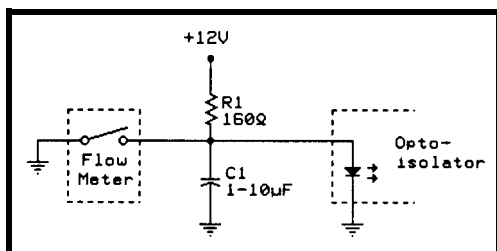


Figure 6-A simple RC circuit can be used to integrate the flow-meter signal, eliminating much of the induced AC noise.

Laplace transforms. This calculation works quite well, but sometimes there is a simpler way which yields the same results.

The problem is with the induced AC waveform, which could be damped out by placing a shunt resistor to ground at the end of the signal lead. Using a shunt resistor would work, except it would have to be very small and have a large current-carrying capacity.

The other alternative is to simply place a DC bias voltage on the signal lead, which gives an offset to the AC signal. A 160-R resistor tied to 12 V provides up to 75 mA of current to drive the LED in the optoisolator. Be aware that the resistor will have to be

able to dissipate up to 0.9 W of heat energy. I recommend using a 5-W resistor to keep the temperatures down.

The requirement of smoothing of the AC waveform itself still remains. A 1-10-μF capacitor can do

this job, providing that you use just enough capacitance to guarantee that the LED in the optoisolator remains biased in the on state when the magnetic reed switch is open. The pull-up resistor and smoothing capacitor form an integrator for the reed switch. This integrating action removes the bounce on the opening and closing of the reed switch.

## CONCLUSION

When you interface counters to real-world devices, you must be aware of the environment you're working in. Most embedded-controller boards provide only basic signal conditioning on their optically isolated inputs. Unfortunately, optically isolated inputs are only fine when all you're looking for is an on/off condition.

But, these inputs can drive you crazy when working with high-speed counters. Sometimes when it gets down to getting dirty, the simpler the solution the better. □

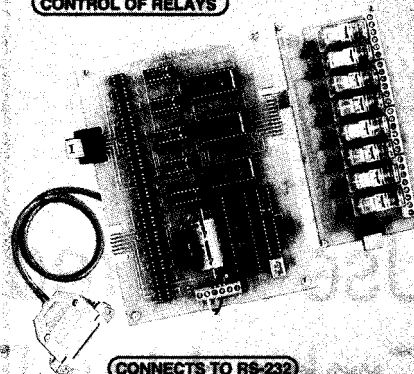
*Bill Payne holds a B.S. in Computing Sciences from the University of Oklahoma, College of Electrical Engineering. He has 12 years experience in the design of computer-based equipment. He holds two semiconductor patents and has four others pending. He is also a Novell Certified Network Engineer (CNE). He can be reached at (918) 224-8615.*

## IRS

- 401 Very Useful
- 402 Moderately Useful
- 403 Not Useful

# RELAY INTERFACE

PROVIDES SOFTWARE CONTROL OF RELAYS

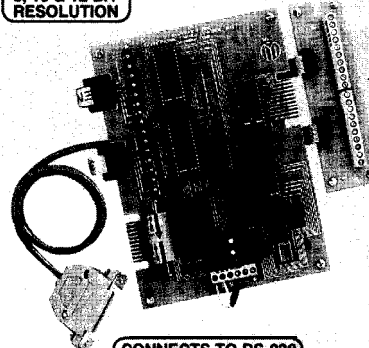


CONNECTS TO RS-232

**AR-16 RELAY INTERFACE (16 channel).....\$ 99.95**  
Two 8 channel (TTL level) outputs are provided for connection to relay cards or other devices (expandable to 128 relays using EX-16 expansion cards). A variety of relays cards and relays are stocked. Call for more info.  
**AR-2 RELAY INTERFACE (2 relays, 10 amp).....\$ 44.95**  
**RD-8 REED RELAY CARD (8 relays, 10 VA).....\$ 49.95**  
**RH-8 RELAY CARD (10 amp SPDT, 277 VAC).....\$ 69.95**

# ANALOG TO DIGITAL

8, 10 & 12 BIT RESOLUTION



CONNECTS TO RS-232

**WC-16 A/D CONVERTER\* (16 channel/8 bit).....\$ 99.95**  
**ADC-8G A/D CONVERTER\* (8 channel/10 bit).....\$ 124.95**  
input voltage, amperage, pressure, energy usage, oysticks and a wide variety of other types of analog signals. RS-422/RS-485 available (lengths to 4,000'). Call for info on other A/D configurations and 12 bit converters (terminal block and cable sold separately).  
**ADC-8E TEMPERATURE INTERFACE\* (8 ch).....\$ 139.95**  
includes term. block & 8 temp. sensors (-40° to 146° F).  
**STA-8 DIGITAL INTERFACE\* (8 channel).....\$ 99.95**  
input on/off status of relays, switches, HVAC equipment, security devices, smoke detectors, and other devices.  
**STA-8D TOUCH TONE INTERFACE\*.....\$ 134.95**  
Allows callers to select control functions from any phone  
**PS-4 PORT SELECTOR (4 channels RS-422).....\$ 79.95**  
Converts an RS-232 port into 4 selectable RS-422 ports  
**CO-485 (RS-232 to RS-422/RS-485 converter).....\$ 44.95**

\*EXPANDABLE...expand your interface to control and monitor up to 512 relays, up to 576 digital inputs, up to 128 analog inputs or up to 128 temperature inputs using the PS-4, EX-16, ST-32 & AD-16 expansion cards.

» FULL TECHNICAL SUPPORT...provided over the telephone by our staff. Technical reference & disk including test software & programming examples in Basic, C and assembly are provided with each order.

HIGH RELIABILITY...engineered for continuous 24 hour industrial applications with 10 years of proven performance in the energy management field.

CONNECTS TO RS-232, RS-422 or RS-485...use with IBM and compatibles, Mac and most computers. All standard baud rates and protocols (50 to 19,200 baud). Use our 800 number to order FREE INFORMATION PACKET. Technical Information (614) 464-4470

24 HOUR ORDER LINE (800) 842-7714'  
Visa-Mastercard-American Express-COD

International & Domestic FAX (614) 464-9656  
Use for information, technical support & orders.

ELECTRONIC ENERGY CONTROL, INC.  
360 South Fifth Street, Suite 604  
Columbus, Ohio 43215-438

# FEATURE ARTICLE

Jeff Fisher

## Use Infrared to Make Embedded Printing Easy

Just how cheap and easy can it be to add a printer to an embedded system? With some nifty innovation, Jeff modifies a HP calculator printer to communicate with his computer using an infrared signal.



i. This is Mr. Bigbucks at Monolithic Megacorp. We really like your Frapdoodle 2000. Work us up a quote on a container-ship load right away. By the way, there's just this one thing...is there some way that it can print a log of fraps on a little printer? Remember, cost is our number one concern!"

"Sure, no problem. I'll have a quote for you in the morning."

Click. Groan. Data logging. Yuk. Let's see, a UART plus glue for serial data (PC board re-layout!), a DE9 connector (case modifications!), a cable to connect to the printer. . . Now, where did I see those surplus cash-register printers advertised? I wonder if I can find a case for them. Oh yea, and a power supply. Groan.

"Where's the aspirin? "

Have you ever needed a little printer for your stand-alone project? We went through this recently and discovered that it is more difficult and expensive than it sounds. But instead of telling you all the reasons you can't do it, I'll tell you how some lateral thinking may achieve the desired results in an inconspicuous way.

### CALCULATING AN APPROACH

You've seen these new personal information managers (PIMs) and high-powered calculators? Many have

optional printers. All these printers have a nonstandard method of connection. Many plug into the calculator or PIM with a custom connector and receive clocked-serial, TTL-level data.

But Hewlett-Packard uses a unique approach. Their HP82240 calculator printer (see Photo 1) receives infrared signals from the calculator. The printer has 24 columns, prints on a 2¼× thermal-paper roll, can operate on internal batteries or a simple external power supply, is available anywhere that sells HP calculators, and is relatively inexpensive (around \$130).

What could be better than a wireless connection? All I need is an infrared LED poking out somewhere, a single bit from the micro to drive it, a little software, and behold the printed word! The interface cost is so low that I could build it into every unit, and then offer the printer as an option.

First, a little reverse engineering. Now, I happen to know as much about infrared as any other human being alive, which is almost enough.

Nobody ever knows "enough" about infrared. But, my tools served me well in deciphering the infrared codes, and I did *eventually* figure out what the heck those extra bits on each character were. Having done the hard work, it turns out that the codes are relatively easy to create.

### ANATOMY OF A CHARACTER

Like most printers, the HP82240 receives eight-bit characters. The

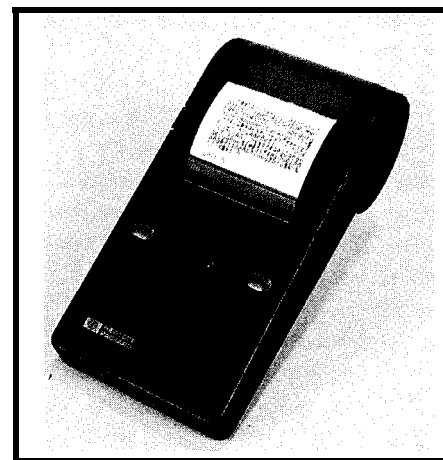


Photo 1—The HP82240B printer—24 columns, graphics capable, battery or AC powered, wireless infrared operation

lower 128 are standard **ASCII** characters. The upper 128 are special characters that include various symbols, foreign characters, accent marks, and so on. Escape codes offer expanded (double-wide) printing, underlining, and even dot-addressable graphics. (This is all explained in the printer's manual.) If you are unfamiliar with infrared, you should read the sidebar on infrared communications before going any further.

It takes 10.92 ms to transmit a character (see Figure 1a). Each character is followed by at least 4.7 ms of no carrier. A carriage return causes the printer to output the line and advance the paper, which takes about 1.8 s. Since there is no feedback from the printer, output must be paced so it doesn't overrun the meager printer buffer. (Note that when the printer is operated on battery power, it runs slower as the batteries discharge.)

Now look at Figure 1b. Each character consists of 13 bits. The start bit (always a one) is followed by four bits of longitudinal-redundancy check (LRC) and eight bits of character. The LRC and character bits are transmitted high bit first. The LRC is calculated using only the byte it is attached to.

There's probably some kind of giant polynomial expression that could be used to create the LRC, but I ignored this. Instead, I created a simple procedure using bit tests and exclusive ORs. (One of the advantages of reverse engineering is that you get to avoid all the theories and math that went into the original design.)

The bits are transmitted using a technique similar to modified frequency modulation (MFM)—that's right, the same scheme used in many disk drives! The advantage of MFM is that it is self-clocking, well understood, and reliable. Note that MFM modulates the frequency of pulses, not the frequency of the carrier. The rules of this peculiar style of MFM are:

- for a data pulse, if the data bit is one, turn the carrier on in the middle of the bit cell. Otherwise, leave the carrier off in the middle of the bit cell.

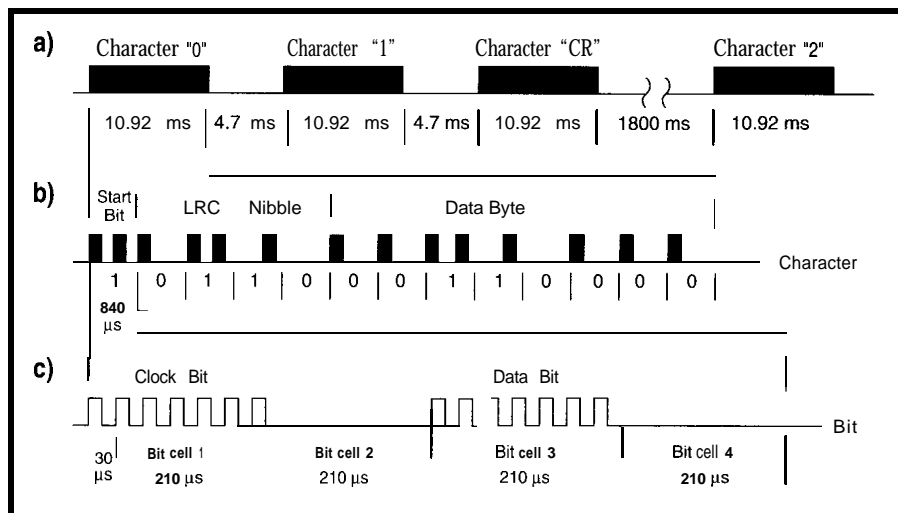


Figure 1—The transmission timing can be successively broken down into finer and finer pieces. Complete characters (a) can be divided into a series of bits (b), which are further described by bursts of IR light (c).

- for a clock pulse, if the previous bit was zero, if it is the start bit, or if the previous bit was the start bit, turn the carrier on at the start of the bit cell.

As you can see in Figure 1c, each bit cell (840 μs) is divided into four pieces, each 210 μs long. The second and fourth piece always have no carrier present. The third piece is the data bit and has a carrier present if the data bit is on. The first piece is where the clock pulse goes, but it is only inserted if the preceding data bit is zero.

The 33 1/3-kHz carrier can run seven cycles in 210 μs, which is just long enough for the detector to trigger and the main reason that the printer only works at desktop distances from the transmitter.

## INFRARED LEDS

I like the SY-IR53L infrared LED available from Radio Shack (276-143). You can drive this LED satisfactorily from many TTL parts as long as you

use an appropriate series resistor to limit the current. Alternatively, you can use some discrete parts to drive the LED closer to its 100-mA rating. Just be aware that stronger is not necessarily better in the bizarre world of infrared. I obtained successful results (over three feet) at only 2.5 mA.

So, all you need to drive a printer from your embedded system is a single TTL output bit and a processor fast enough to turn the bit on and off every 15 μs. If you can't dedicate your processor to driving the LED during printing and 15 μs is too fast to handle with interrupts, you can add a 33 1/3-kHz oscillator, which you enable and disable on 210-μs intervals. The frequency doesn't have to be precise, so a 555 or ceramic oscillator is adequate.

## A SIMPLE EXAMPLE

So much for theory. Now to really do something.

For this example, we used an IBM PC-compatible computer. We connect the infrared LED and 1-kΩ series resistor to data bit zero of the parallel port (see Figure 2). Most PCs use an octal latch such as the 74LS374 for the parallel port's data bits. This part can source up to 2.6 mA, which is adequate for this example. The hardware is that simple! Software in the PC can now drive the infrared LED.

The example program, `printp` (Listing 1), is written in Turbo C. It reads input and sends

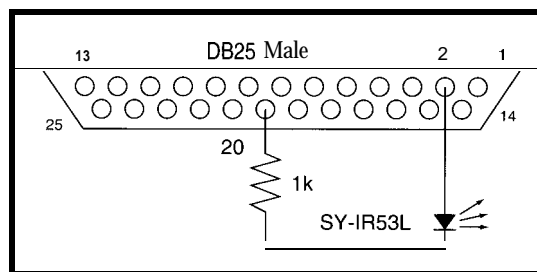


Figure 2—The only hardware required to send data to the infrared printer is a resistor and an LED connected to a bit on the PC's printer port. Software does all the work.



# \$99 4A C Compiler?

You heard right. A quality C compiler designed for the 8051 microcontroller family, just \$99, including the Intel compatible assembler and linker. And a source level simulator is also available, just \$79. A great companion to our fine Single Board Computers, like those below. CALL NOW!

## 552SBC

**80C552** a '51 Compatible Micro  
40 Bits of Digital I/O  
8 Channels of 10 Bit A/D  
3 Serial Ports (RS-232 or 422/485)  
2 Pulse Width Modulation Outputs  
6 Capture/Compare Inputs  
1 Real Time Clock  
64K bytes Static RAM  
1+ UVPROM Socket  
512 bytes of Serial EEPROM  
1 Watchdog  
1 Power Fail Interrupt  
1 On-Board Power Regulation

Priced at just \$299 in single quantities. Call about our 552SBC C Development Kit, just \$449.

## 100 MHz 8051!

Our popular 8031SBC can now be shipped with Dallas Semi's hyperactive **DS80C320**, an 8051 on steroids. Averaging 3x faster than the standard 51, your project can really scream! **Call or ftp for pricing and brochures today!**

Other versions of the 8031SBC have processors with on-chip capture registers, EEPROM, IIC, A/D and more. Call or ftp for a list!

**8031SBC** as low as \$49

Call for your custom product needs. Quick Response.

**HTE** HiTech Equipment Corp.  
9400 Activity Road  
San Diego, CA 92126  
[Fax: (619) 530-1458]

Since 1983

(619) 566-1892



Internet e-mail: info@hte.com  
Internet ftp: ftp.hte.com

## INFRARED COMMUNICATIONS

The most confusing part about infrared data transmission is that there are three frequencies involved.

The highest frequency, more appropriately called *wavelength*, is the infrared light emitted by the LED. For most consumer remote controls (and this calculator printer), the wavelength is around 880 nm. This wavelength is in the *near infrared* band-somewhere between visible light and actual heat radiation. You should choose an infrared LED that emits in a range of 850-900 nm.

The next highest frequency is that of the *carrier*. The carrier more reliably detects the infrared turning on and off at a known frequency than trying to detect a steady state on or off. This makes sense since *everything* emits infrared energy, so the background level is constantly going up and down. Typical carrier frequencies range from 20 kHz to 80 kHz with 40 kHz being most common. Since the receiver circuits are pretty simple, the carrier can often vary by as much as 20% and still works.

The next lowest frequency is used for the pulses that carry the actual information. Most infrared data is transmitted by turning the carrier on and off at times determined by the data. This method results in pulses of the carrier, followed by no carrier, and is called *Pulse Code Modulation* (PCM).

Both the on and off times can vary to carry data, so the frequency of the pulses is not necessarily constant. Usually, only the on or off time is varied to keep things simple. Since it may take many cycles for the receiver to react to the carrier, the on times are usually at least ten times longer than the carrier wavelength. Thus, the pulses are usually measured in hundreds of microseconds of on and off time.

Listing 1--This sample program demonstrates printing on the HP82240B infrared printer. It reads from standard input.

```
#include <stdio.h>
#include <dos.h>

/***** STATIC VARIABLE DECLARATIONS *****/
#define LPT1      0x378
#define LPT2      0x278
#define LPT       LPT1 /* which printer port to use */
#define THIRTY    6 /* adjust to get 33.3-kHz carrier */
#define TWOTEN   55 /* adjust to get 210 µs delay */

static unsigned short counter;

/XX*****/
static void printhp(char);
static void wiggle(void);
static void mydelay(void);

/*****\
Main function
*****/
int main(int argc, char *argv[])
{
    char c;
    delay(0); /* initialize the delay system */
    outp(LPT + 2, 0); /* enable output on parallel port */

    while ((c=getchar()) != EOF){
        printf("%c", c);
    }
}
```

(continued)

Listing 1-continued

```

    printhp(c);
    delay(5);          /* intercharacter gap */
    if (c == '\n')
        delay(1800; /* add about 1.8 seconds delay after CR */

    return 0;

/*****\
    Print a single character on the HP infrared printer
    \*****/
void printhp(char c)

    unsigned int i, j;
    static int lastbit;

    i = c;              /* put in the character */
    if (i & 0x01) i ^= 0x300; /* put in the check nybble */
    if (i & 0x02) i ^= 0x500;
    if (i & 0x04) i ^= 0x600;
    if (i & 0x08) i ^= 0x900;
    if (i & 0x10) i ^= 0xa00;
    if (i & 0x20) i ^= 0xc00;
    if (i & 0x40) i ^= 0xe00;
    if (i & 0x80) i ^= 0x700;
    i |= 0x1000;        /* put in the start bit */

/* rotate it all out: start-bit, check-nybble, char */
for (j = 0, lastbit = 0; j < 13; j++, i <<= 1){
    if (lastbit)
        mydelay();          /* no clock pulse needed */
    else
        wiggle();          /* clock pulse */
    mydelay();              /* wait until middle of bit cell */
    if (i & 0x1000)
        wiggle();          /* on bit */
    else
        mydelay();          /* off bit */
    mydelay();              /* fourth part of bit cell */
    if (j)
        lastbit = i & 0x1000; /* save bit for clocking next pass */

/*****\
Delay for 210 μs
\*****/
static void mydelay(void)

    counter = 0;
    while (counter++ < TWOTEN);

/*****\
Wiggle the output bit 7 times to create a carrier
\*****/
static void wiggle(void)
{
    int i;
    for (i = 0; i < 7; i++){
        counter = 0;
        outp(LPT, 0xff);          /* turn the bit on */
        while (counter++ < THIRTY/Z); /* wait 15 μs */
        outp(LPT, 0x00);          /* turn the bit off */
        while (counter++ < THIRTY); /* wait 15 μs */
    }
}

```

each character to the printer. It is meant to be used as filter in a command-line pipe (e.g., type `a_f i l e | p r i n t . h p`). The main routine handles the reading of characters and gross timing issues. The routine `p r i n t h p` encodes and outputs the characters. The `mydelay` routine pauses for 210 μs, and `wiggle` creates a brief 33.333-kHz carrier pulse.

If you want to run `printhp`, you need an oscilloscope or logic analyzer to calibrate the two routines. First, get the carrier cycle length as close to 30 μs as possible by changing the `THIRTY` definition. Then, adjust `TWOTEN` so the gap between the first two carrier pulses is as close as possible to 210 μs. This should get your printer working.

## THE NEXT DAY

"Mr. Bigbucks again. Thanks for the quote and the demonstration of your new FrapLog option last week. But, uh, about the order.. . Monolithic Megacorp was just bought out by Polyolithic Gigacorp. It seems that Polyolithic just don't give a frap.. ."

Click. Oh, well. At least some of our real customers will appreciate it. □

*Jeff Fisher is president of HomeTech Solutions, a home automation manufacturer and retailer in San Jose, California. He may be reached at (408) 257-4406 or 71431.3343@compuserve.com.*

## SOURCES

The HP82240B printer is available from:

HomeTech Solutions  
10570 S. De Anza Blvd.  
Cupertino, CA 95014  
(408) 257-4406  
Fax: (408) 257-4389

Hewlett-Packard  
Portable Computer Division  
1000 NE Circle Blvd.  
Corvallis, OR 97330  
(503) 757-2000

## IRS

404 Very Useful  
405 Moderately Useful  
406 Not Useful

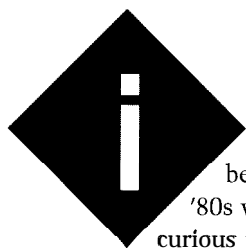
# It's Not Just for Memory Anymore

## An Introduction to PCMCIA

You know the credit-card revolution has made a significant impact when you look at laptop peripherals today. Lalo's article introduces us to this evolving breed of devices.

### FEATURE ARTICLE

Lalo J. Gastriani



It was around the beginning of the '80s when a few curious musicians started wondering about those "funny little connectors" found on the back panel of high-end synthesizers and sampling keyboards. Accessory catalogs were studied; inquiries were made.

Soon, they learned that expansion cards could be obtained and plugged into those sockets. By issuing the correct commands, keyboard parameter sets or "patches" as well as waveform data could be saved to the small memory cards and recalled later. The fact that these cards could be inserted or extracted while the unit was running made them even more attractive.

The offspring of those funny little sockets have now found their way onto virtually every notebook computer made today. The devices, which plug into these sockets, grew out of the JEIDA (Japan Electronics Industry Development Association) standard and are now known as *personal computer cards* or simply *PC cards*. These credit-card-sized peripherals contain anything from real-time global positioning systems to tiny hard drives with rotating media. Even more prevalent now are high-speed modems, which enable people on the road to "jack in" to that Infobahn we keep hearing about.

The computer industry is just beginning to see the benefits and potential of PC cards. For about the last three years, the standard for these devices and the system software that controls them, collectively referred to as PCMCIA (Personal Computer Memory Card International Associa-

tion), have quickly become more known to both manufacturers of computers and end-users alike. As it now rolls into its latest 3.0 incarnation, the specification has endured the test of time and public scrutiny.

After a slow start, great strides have been taken in discovering the essential criteria which enable the cards to operate on a particular host and also provide the foundation for successful real-world, cross-platform operation. Although we're not there yet, this achievement will provide us with what is known in the industry as *interoperability*.

Already, we are at the point that a notebook computer is considered stripped down if it doesn't contain at least one PCMCIA socket. What's more, PCMCIA cards are sold everywhere from giant, computer-warehouse stores in your neighborhood to the shop-at-home cable channels. Consumers are asking manufacturers of desktop systems to provide PC-card capability as part of their solution on subnotebooks as well as desktops. Perhaps, we may soon see sockets on everything from automobiles to public telephones.

In the embedded world, PCMCIA sockets and cards offer a whole new breed of device. Designers can simply add a socket or two to their designs along with the proper enabling firmware, and flexibility and future expansion is possible.

But, let's take a closer look at PCMCIA.

#### HARDWARE

Sockets are the basic receptacle PC cards are inserted into and removed from. The socket connector itself normally rests inside the host computer and consists of a plastic housing with a double row of 34 pins giving a

Type	Thickness (mm)
I	3.3
II	5
III	10.5
IV	13.5

Table 1—Since different functions require different amounts of physical space, four card thicknesses are defined. Type IV is a nonstandard form factor, which PCMCIA rejected, but still is used by some manufacturers.

Photo 1—A Type III PCMCIA card typically houses the rotating media of a hard drive.



total of 68 pins, each spaced 1 mm apart.

Photo 1 and Figure 1 illustrate the pin layout in a PCMCIA card end. The exact pinout of the socket can be found in 4.2-4.3 of *PCMCIA PC Card Standard Release 2.01*.

Cards are inserted into and extracted from the sockets while the host's system power is active, a technique also known as *hot swapping*. Hot swapping and the whole notion of temporal devices in personal computers is a fairly new concept and as such has its own unique set of problems. These problems center mainly on the fact that, until now, system resources were allocated to devices present at system bootstrap. The devices stayed present throughout the duration of the computing session, remaining nearly entirely static until the session was terminated.

Plugging a PC card into a powered-up system and expecting the functionality of the card to suddenly become active is quite an expectation. How this is done is where the real magic of technology lies.

If you were to look crosswise at the socket pins, you'd see that some of the pins do not protrude as far as others. The power rails are located on the outermost pins and are the longest of the pins. When a card is inserted, they make contact first. Similarly, when it is extracted, they retain the longest. This power arrangement enables buses to be powered up and tristated when (or very soon after) the card is inserted.

In addition to data and address lines, there are also control lines, battery indicators, a single interrupt line, and a card-detect line, which can indicate card insertions and extractions.

To be able to insert a modem card,

have the OS recognize and configure it, spawn a communication program, connect to an on-line service, down-

load E-mail, and then disconnect is an amazing series of events. To make the scenario even more interesting, remove the modem card and insert a 120-MB, rotating hard drive containing a customer database, which launches an invoice program.

The interface between the socket's 68-pin bus and the host is known as the *socket controller*. It manages the low-level aspects of the socket (i.e., power, interrupt routing, memory and I/O window allocation, etc.) according to program control.

On desktop and notebook systems, the socket controller is typically a dedicated chip which is part of some

Memory Only Card Interface: Always available at card insertion				Memory Only Card Interface: Always available at card insertion			
Pin	Signal	I/O	Function	Pin	Signal	I/O	Function
1	GND		Ground	35	GND		Ground
2	D3	I/O	Data bit 3	36	CD1	O	Card detect
3	D4	I/O	Data bit 4	37	D11	I/O	Data bit 11
4	D5	I/O	Data bit 5	38	D12	I/O	Data bit 12
5	D6	I/O	Data bit 6	39	D13	I/O	Data bit 13
6	D7	I/O	Data bit 7	40	D14	I/O	Data bit 14
7	CE1	I	Card enable	41	D15	I/O	Data bit 15
8	A10	I	Address bit 10	42	CE2	I	Card enable
9	OE	I	Output enable	43	RFSH	I	Refresh
10	A11	I	Address bit 11	44	RFU		Reserved
11	A9	I	Address bit 9	45	RFU		Reserved
12	A8	I	Address bit 8	46	A17	I	Address bit 17
13	A13	I	Address bit 13	47	A18	I	Address bit 18
14	A14	I	Address bit 14	48	A19	I	Address bit 19
15	WE/PGM	I	Write enable	49	A20	I	Address bit 20
16	RDY/BSY	O	Ready/Busy	50	A21	I	Address bit 21
17	V <sub>cc</sub>			51	V <sub>cc</sub>		
18	V <sub>pp1</sub>		Programming Supply Voltage 1	52	V <sub>pp2</sub>		Programming Supply Voltage 2
19	A16	I	Address bit 16	53	A22	I	Address bit 22
20	A15	I	Address bit 15	54	A23	I	Address bit 23
21	A12	I	Address bit 12	55	A24	I	Address bit 24
22	A7	I	Address bit 7	56	A25	I	Address bit 25
23	A6	I	Address bit 6	57	RFU		Reserved
24	A5	I	Address bit 5	58	RESET	I	Card reset
25	A4	I	Address bit 4	59	WAIT	O	Extend bus cycle
26	A3	I	Address bit 3	60	RFU		Reserved
27	A2	I	Address bit 2	61	REG	I	Register select
28	A1	I	Address bit 1	62	BVD2	O	Battery voltage detect 2
29	A0	I	Address bit 0	63	BVD1	O	Battery voltage detect 1
30	D0	I/O	Data bit 0	64	D8	I/O	Data bit 8
31	D1	I/O	Data bit 1	65	D9	I/O	Data bit 9
32	D2	I/O	Data bit 2	66	D10	I/O	Data bit 10
33	WP	O	Write protect	67	CD2	O	Card detect
34	GND		Ground	68	GND		Ground

I/O and Memory Card Interface: Available only after socket has been put into I/O mode			
Pin	Signal	I/O	Function
16	IREQ	O	Interrupt request
18	V <sub>pp1</sub>		Programming & Peripheral Supply
33	IOIS16	O	IO Port is 16-bit
44	IORD	I	IO Read
45	IOWR	I	IO Write
52	V <sub>pp2</sub>		Programming & Peripheral Supply 2
60	INPACK	O	Input Port Acknowledge
61	REG	I	Register select and IO Enable
62	SPKR	O	Audio Digital Waveform
63	STSCHG	O	Card Statuses Changed

Figure 1—The 68-pin PCMCIA interface between a PC card and its socket shows the signal definitions in memory mode. The lower table describes those pins whose functions change when the socket is in I/O mode. The mode is determined by the state of the \*REG signal.



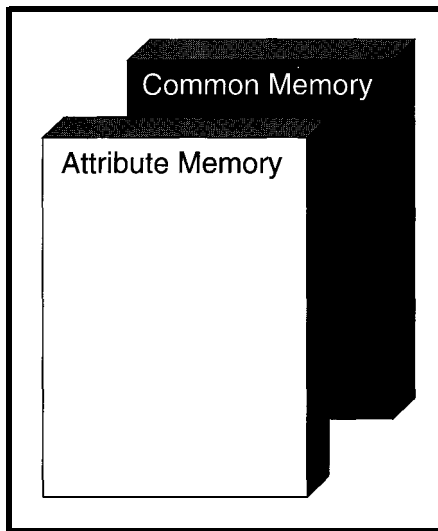


Figure 2—Common and attribute memory share the same address space. The client selects which one is active.

sort of host adapter. The Intel 82365SL and the Databook 86082 are examples of popular socket controllers.

In embedded systems, the socket controller can be implemented with discrete logic, as one of the standard pieces of silicon, or as a single task running along with other tasks in the microprocessor. As another possible route, you could use a dedicated microcontroller like the PIC.

## RESOURCES

As I mentioned, the key to enabling PC cards in a system is to provide them the resources they require. However, before I talk about how a card tells the system what resources it needs, let's take a look at the various categories of resources available on a host.

Power is the most obvious and most basic resource. It is provided to the card via the power rails and is typically 3.3 or 5 V, depending on the host system. A typical socket controller may be requested to provide higher voltages for certain cards (e.g., flash-memory cards, which usually require 12 V for programming). To switch the power rails to the card, the socket controller either has built-in active switches or external MOSFET devices.

To interface a particular card's function to the rest of the system, memory windows are created which map a particular region of the card's address space into the host system's

address space. The exact place within the host map is determined by system software. The size of the memory window as well as its attributes are usually also programmable.

On systems with I/O space, a card's I/O decoding might require connection to a particular range of I/O locations. This is particularly critical when a PC card is set up to emulate a static device.

Another important resource sometimes applicable is the PC card's interrupt routing. Refer again to Figure 1. Notice that a card has only a single IREQ line which can be routed through the socket controller to any host interrupt line. This routing is usually programmable, allowing for flexible compatibility.

The IREQ signal from the card should not be confused with the card-detect interrupt from the socket, which is used for insertion and extraction detection and has nothing to do with the card's IREQ signal. Instead, the IREQ line is to be used by the card in a card-specific manner.

While the currently released PCMCIA specs do not include support for DMA to and from a PC card, the

fourth type of card, which they call Type IV. Although the Type-IV form factor was rejected by the PCMCIA committee, the 13.5mm slot can be found on some Toshiba models. Table 1 shows the correlation between thickness and type.

## MEMORY SPACE

As Figure 2 illustrates, there are two types of memory space on a PC card: common and attribute. Even I/O cards must implement attribute memory for this is where the card's CIS (card information structure) is. The CIS contains the card's "biography" as well as its "wish list" for system resources.

By definition, when a card is powered up, it initializes in memory mode. This ensures that as each card is inserted, it begins in a known state. This state can best be characterized as:

- memory mode active [as opposed to I/O mode active]
- attribute memory active (as opposed to common memory active)

Attribute and common memory share the same address space (64 MB), but only one can be active at a time. The

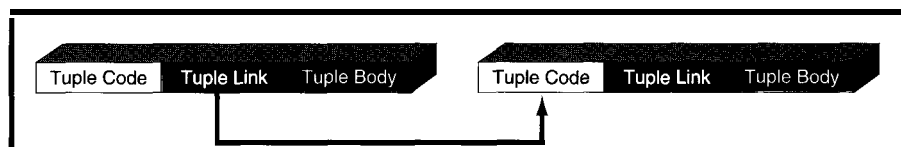


Figure 3—Every card has information built into it detailing its manufacturer, part number, and other items. The information is conveyed to the host through the use of tuples, which form the basis for a linked list of data objects

next release (version 3.0) accommodates it. Socket controllers are already appearing which have the ability to provide DMA channels to and from the card's address space. Until the DMA functionality of the PCMCIA software layers are standardized, DMA implementations are ad hoc.

## CARD FORM FACTORS

There are currently three types or configurations of PC cards recognized by the standards: I, II, and III. The primary difference between the types is the thickness of the card.

The Japanese consortium of card standards (JEIDA) has taken a small lead in this area having designated a

socket controller provides the means to select one of the two memory spaces.

Attribute memory is typically (although not always) nonvolatile, contains the CIS, and if applicable, the configuration registers. Attribute memory must begin at offset 0, but need not be in a single contiguous region. Most cards implement their CIS and configuration registers at fairly low offsets in attribute space (according to PCMCIA guideline), which ensures addressability by all hosts. This also minimizes that amount of page mapping required by the system software to access these regions.

A quirk of attribute space is that it is a 16-bit interface only. In other words, when reading CIS information or writing configuration registers, only the even bytes are considered valid data. Likewise, on write cycles, the card is only obliged to handle writes to the even bytes.

Mass storage PC cards have a separate memory space called common memory, which provides the main write/read memory space for the card. In a 4-MB SRAM card, for instance, common memory is viewed as a 4-MB linear region. By setting up a mapping window and by manipulating the offset addressed by this window into common memory, any byte contained on the card can be accessed.

Common memory is treated as an S-bit array of bytes. Both the even and odd bytes are valid.

### CARD INFORMATION STRUCTURE (CIS)

The Card Information Structure is implemented in a format known as the *metaformat*. As mentioned, the CIS is the card's biography, which contains, among other things, the:

- manufacturer
- part number
- voltage and current requirements
- absolute maximum ratings
- one or more configuration scenarios (I/O cards only)

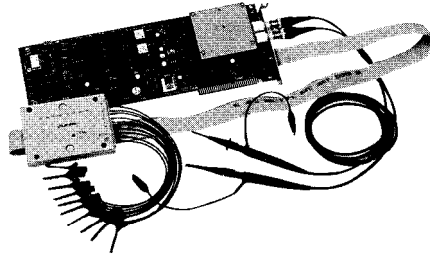
In theory, when a card is inserted, system software powers up the card and starts looking at the CIS. As the CIS is traversed and parsed, more information about the card may be obtained.

This information is conveyed to the client through the use of *tuples*. Tuples form the basis for a linked list of data objects which provide enough information about the card that it may be properly enabled and used. Figure 3 presents the layout of a tuple.

As you can see, layer 1 offers the basic compatibility tuples. The first byte contains the tuple code and may fall into the range of 00-1D (hex). The values between 01E and 3F are reserved for future expansion. Layer 2 is for data recording format and lies

## PC-Based Instruments 200 MSa/s DIGITAL OSCILLOSCOPE

**HUGE BUFFER  
FAST SAMPLING  
SCOPE AND LOGIC ANALYZER  
C LIBRARY W/SOURCE AVAILABLE  
POWERFUL FRONT PANEL SOFTWARE**



**\$1799 - DSO-28204 (4K)**  
**\$2285 - DSO-28264 (64K)**

### DSO Channels

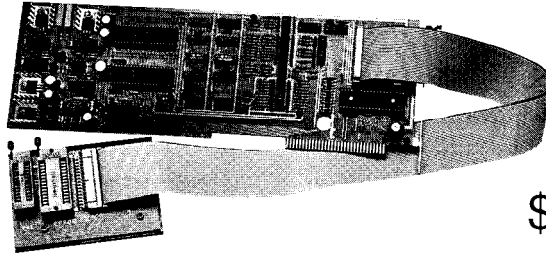
2 Ch. up to 100 MSa/s  
or  
1 Ch. at 200 MSa/s  
4K or 64K Samples/Ch  
Cross Trigger with LA  
125 MHz Bandwidth

### Logic Analyzer Channels

8 Ch. up to 100 MHz  
4K or 64K Samples/Ch  
Cross Trigger with DSO

## Universal Device Programmer

PAL  
GAL  
EPROM  
EEPROM  
FLASH  
MICRO  
PIC  
etc..

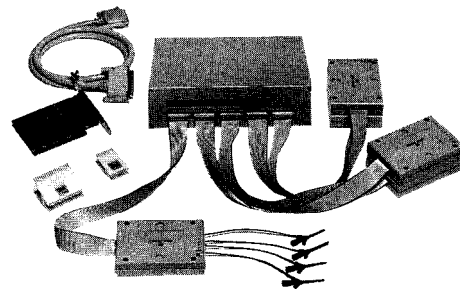


**\$475**

Free software updates on BBS  
Powerful menu driven software

## 400 MHz Logic Analyzer

up to 128 Channels  
up to 400 MHz  
up to 16K Samples/Channel  
Variable Threshold Levels  
8 External Clocks  
16 Level Triggering  
Pattern Generator Option



**\$799 - LA12100 (100 MHz, 24 Ch)**  
**\$1299 - LA32200 (200 MHz, 32 Ch)**  
**\$1899 - LA32400 (400 MHz, 32 Ch)**  
**\$2750 - LA64400 (400 MHz, 64 Ch)**

Call (201) 808-8990



**Link Instruments**

369 Passaic Ave, Suite 100, Fairfield, NJ 07004 fax: 808-8786

between 40 and 45, Layer 3 provides data organization and is in the range of 46–7F, and Layer 4 offers system-specific features and is found in 80–FE. The special value FF present as either a tuple code or a link marks the end of the tuple list. Table 5.2 of section 5.7 of the PCMCIA PC Card Standard describes all tuple codes and their meaning.

This scheme is quite flexible and allows for simple or complex CIS structure. Special tuples (long links) allow the CIS to span attribute and common memory space, thereby enabling elaborate CIS schemes to be implemented.

## CARD FUNCTIONS

Memory cards are the simplest of PC cards to configure. They typically require only power for reading, writing, and mapping a window. In the case of nonvolatile memory like flash, special programming of voltage and current requirements can be found in the card's CIS.

I/O cards are the most prevalent of PC cards. By far the most popular I/O card is the modem or fax/modem card. I/O cards may or may not require I/O windows to be set up for them through the socket controller. When I/O is needed, only certain I/O locations are allowed to be decoded by the card.

This technique enables a PC card to look like any device that would normally be attached to a particular host's bus. By setting up the proper I/O windowing, the I/O card's register set can be made to respond to host bus requests within certain ranges.

Some I/O cards have memory-mapped registers, which are present in either the card's common or attribute memory space. By supporting this kind of operation, a particular card can ensure interoperability (or at least a higher chance of achieving it) by working in systems that have I/O space (e.g., the Intel x86 series) or those that do not (e.g., the Motorola 68k series).

All cards, when first powered up, assume memory mode. This facilitates the reading and parsing of the card's CIS. The client selects a particular configuration based on information in

the CIS and a configuration scenario which makes sense for the host at that particular time. After configuration is selected and set, an I/O card then assumes I/O mode. The changeover at the socket level between memory and I/O mode causes a few pins to be redefined. Consult Figure 1 for the differences in pin definitions between memory and I/O mode.

Lately, some of the newer cards coming out have two or more separate and distinct functions. For example,

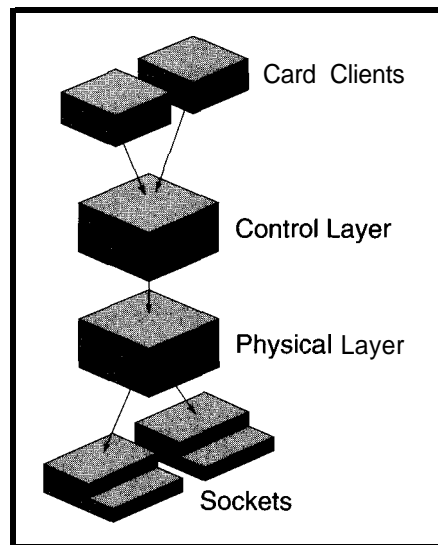


Figure 4—PCMCIA software is broken into several layers which apply mostly to desktop and notebook systems. By using separate layers, it's possible to have a common interface at the top and very different physical devices at the bottom.

Xircom has developed an Ethernet/fax-modem card, which combines a miniature Ethernet adapter and a high-speed fax/modem. Through this type of arrangement, both of the card's functions share the same socket.

This multifunctionality poses some challenges in the area of interrupt routing since there is only one IREQ pin from the card. This one IREQ pin, which is routed to one of the host's interrupt lines, must be able to signify an interrupt from two or more separate sources.

Some new proposals are being discussed by PCMCIA which include the use of virtual sockets to "house" the second and subsequent functions on a particular card. However, at this time, there have been no ratified extensions or enhancements to the specifications for multifunction cards.

Still another class of PC cards are those that appear to the system to be a standard peripheral known as an AT attachment or an ATA device. These cards may contain solid-state memory or rotating media. When the card's resources are properly configured, the card appears as a hard drive.

In a typical desktop or notebook application, these devices are connected to the host file system by a device driver and appear as another logical device. In an embedded system, an ATA device could contain the operating system (with a virtual memory-storage partition) and/or extra storage.

Many ATA devices contain programmable watchdogs. These enable the power management aspects of a particular OS to set timeout values for spinning down the drive during periods of inactivity.

## SOFTWARE

Whether the environment is desktop, notebook, or embedded, the general intent of PCMCIA software at large is universal:

- enable cards within sockets
- protect systems from bad cards when possible
- encapsulate the details of socket electronics
- allow many clients of sockets to coexist
- manage resources
- provide interoperability and compatibility

How these goals are best achieved is almost entirely platform specific. But, the basic tenets are the same. By breaking up the software into layers and by defining each layer's boundaries and the part that layer serves greatly simplifies the conceptual visualization of implementing solutions.

PCMCIA software is broken up into two basic areas: clients and servers. The server layers provide the services to the socket controller. Clients, on the other hand, interact with the servers and other clients in an effort to configure the cards and keep them running.

Layers, outlined in the PCMCIA

documents under socket services and card services, apply mostly to desktop and notebook systems. Figure 4 presents the various layers which make up PCMCIA software.

The physical layer controls the hardware registers or other programming model of the socket controller(s) on a particular platform. In essence, this layer reports the capabilities of each socket controller and enables the encapsulated manipulation of them.

A simple interface to higher-level layers allows these higher layers to deal with logical sockets. The PCMCIA standard calls this layer **socket services**. There may be one or more of them in a system, each handling one or more socket controller(s).

Above the physical layer lies the control layer which reacts to events, manages resources, and interacts with the card clients in a system. This layer is part of the operating system and contains further abstractions that, at this level, enable memory technologies to be encapsulated according to their topology and power requirements.

Each client makes resource requests to this layer. Each request is weighed against collisions with other system components; collisions cause rejections. This layer is called **card services** in PCMCIA's standards. There is only one case of it in any system.

**Clients** are tasks at the top of the PCMCIA food chain. When a card is inserted, the ensuing interrupt is handled by the physical and control layer. This eventually causes an insertion event or message to be broadcast to all clients registered with the control layer.

At this point, a client typically parses the card's CIS looking for either recognizable "landmarks" within the manufacturer's ID tuple or moving right into the configuration tuples. Once it starts configuration, it constructs resource requests and passes them off to the control layer.

This process continues until a particular configuration scenario is accepted by the control layer, at which time it is latched and the card is enabled. The card remains in this state until it is either extracted or some sort of power-management message is

broadcast, forcing the card and socket into a lower-power state.

Extracting the card reverses the process so that resources are freed and given back to the host.

## CONCLUSION

PCMCIA technology opens doors in many areas of desktop, notebook, and hand-held computing. This introduction provides a foundation for further study. In a follow-up article, I will focus more intensely on embedded systems, detailing an actual application of PCMCIA. 📧

**Lalo J. Gastriani is a consulting engineer based in Southern California. Since 1982, he has concentrated on real-time software architecture and design. Since 1990, he has been involved in PCMCIA. Lalo may be reached at [ljg@kaiwan.com](mailto:ljg@kaiwan.com) or P.O. Box 80801, Rancho Santa Margarita, CA 92688.**

## CONTACTS

PCMCIA specifications:  
 Personal Computer Memory  
 Card International Assoc.  
 (PCMCIA)  
 1030G East Duane Ave.  
 Sunnyvale, CA 94086  
 (408) 720-0107  
 Fax: (408) 720-9416

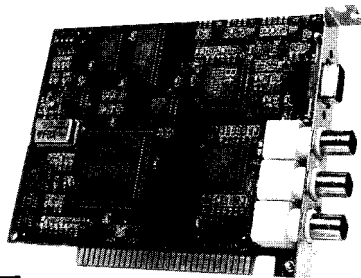
PCMCIA peripherals:  
 Integral Peripherals, Inc.  
 5775 Flatiron Parkway, Ste. 100  
 Boulder, CO 80301  
 (303) 449-8009  
 Fax: (303) 449-8089

Xircom  
 26025 Mureau Rd.  
 Calabasas, CA 91302  
 (818) 878-6423  
 Fax: (818) 878-7630

## IRS

407 Very Useful  
 408 Moderately Useful  
 409 Not Useful

# PRECISION FRAME GRABBER FOR ONLY \$495\*



Introducing the CX100 precision video frame grabber for OEM, industrial and scientific applications. With sampling jitter of only  $\pm 3$ ns and video noise less than one WB, ImageNation breaks new ground in imaging price/performance. The CX100 is a rugged, low power, ISA board featuring rock solid, crystal controlled timing and all digital video synchronization. A Software developers will appreciate the simple software interface, extensive C library and clear documentation. The CX100 is a software compatible, drop-in replacement for our very popular Cortex I frame grabber. A Call today for complete specifications and volume pricing.

**ImageNation Corporation**  
 Vision Requires Imagination  
 800-366-9131

P.O. BOX 276 BEAVERTON, OR 97075 USA PHONE (503) 641-7408 FAX (503) 643-2458 BBS (503) 626-7763

### — CX100 FEATURES —

- Crystal Controlled Image Accuracy
- . Memory Mapped, Dual-Ported Video RAM
- Programmable Offset and Gain
- . Input, Output and Overlay LUTs
- Resolution of **5 12x486** or Four Images of 256x243 (CCIR 512x512 & 256x256)
- . Monochrome, 8 Bit, Real Time Frame Grabs
- . Graphics Overlay on Live or Still Images\*\*
- External Trigger Input
- . RGB or B&W, 30 Hz Interlaced Display
- . NTSC/PAL Auto Detect, Auto Switch
- . VCR and Resettable Camera Compatible
- . Power Down Capability
- BNC or RCA Connectors
- . Built-In Software Protection\*\*
- 63 Function C Library with Source Code
- . Text & Graphic Library with Source Code
- . Windows DLL, Examples and Utilities
- . Software also available free on our BBS
- . Image File Formats: GIF, TIFF, BMP, PIC, PCX, TGA and WPG

\*\* THESE OPTIONS AVAILABLE AT EXTRA COST

\* \$495 IS DOMESTIC, OEM SINGLE UNIT PRICE.

# FEATURE ARTICLE

Tracey Lee  
& Kok-Leong Ong

## Speeding and Slimming Your Port Access

### A Different Way of Reading from the PC Parallel Port

and connectors have to be one bit wider.

To accomplish this variation, software handling the transfer needs to generate a READ signal using one of the parallel port pins. We explored the possibility of using this signal to indicate the order of nybble read. Assuming a negative logic READ, our solution was to use the low level (active) to indicate one nybble and to use the high level (inactive) to latch the data and gate the next nybble to the interface.

Thus, we save the two instructions needed to manipulate the nybble indicator bit in Steve's approach. We also save on one extra connection and do not have to perform surgery on the parallel card as suggested by Steve.

By the way, although we use the DDT-5 1 as our example, we understand that it has been discontinued as a product. The techniques described in this article, however, can be applied to parallel ports in general or with other devices that interface to the PC in a similar manner.

#### THE PARALLEL INTERFACE

Before getting into the details of the solution, let's take a look at the PC parallel-port interface. Enhancements to the interface starting from the PS/2

While many PC manufacturers today are supplying their machines with true bidirectional parallel ports, reading data in with older machines is still tricky. Here's one method that requires no changes to the PC.



When we came across the DDT-5 1 805 1 emulator project presented by Steve

Ciarcia several years ago (BYTE 1988), we decided to improve on his implementation. The DDT-51 communicates with the PC using its parallel printer port. Steve modified a standard port for bidirectional operation, but we wanted to use an unmodified port. We decided to forego the soldering iron in favor of a software solution.

We chose to read data a nybble at a time, using one of the other parallel port pins to indicate which of the two nybbles transferred. Of course, the software later combines both nybbles. This method slows down data transfer, especially if the software is written using a high-level language, and uses one extra pin of the port interface. What is more important, however, is that the port and device have to make that one extra connection. This means that cable

PC Address	Bit No.	Corresponding Pin Name	Pin No. DB-25	I/O Direction	
Base	0	D0	2	Output Only	
	1	D1	3		
	2	D2	4		
	3	D3	5		
	4	D4	6		
	5	D5	7		
	6	D6	8		
Base + 1	7	D7	9	Input Only	
	0	Not Used	NA		NA
	1				
	2				
	3	-Error	15		Input Only
	4	SLCTD	13		
	5	PE	12		
6	-ACK	10			
7	Busy	11			
Base + 2	0	Strobe	1	Input and Output	
	1	Auto FD XT	14		
	2	-Init	16		
	3	SLCT IN	17		
	4	IRQ Enable		NA	
	5				
	6	Not Used	NA		
	7			NA	

Table 1 --The functions on original PC parallel port interface are accessed through three consecutive I/O ports.

Listing 1—This sample code reads in a byte of data using the NYB\_IND technique. If pin 16 is used as NYB\_IND, then pin 17 becomes the RD pin and both pins are negative logic.

```

Port [$37A] := $00;           {Read first nybble}
FirstNibble := (Port [$379] And $F0) SHR 4;

Port [$37A] := $08;         {Get next nybble}

SecondNibble := Port [$379] And $F0;
Port [$37A] := $0C;         {Pull RD high}

ByteData := FirstNibble Or SecondNibble; {Combine the nybbles}

```

series of computers have enabled bidirectional S-bit data transfer. Recently, a DMA (direct memory access) function was included as well!

However, we will restrict ourselves to the “original” parallel port, which consists of a 25-pin D connector (for pin assignments, see Table 1). Each port has three addresses associated with it in the I/O map of the PC. The first parallel port, or LPT1, has a base address of 378H, so its addresses would be 378H, 379H, and 37AH. Each address allows us to read or write certain data to the parallel port to control the logic state of each pin.

Due to the design of the standard parallel port, certain pins have negative logic while others use positive logic. Port 379H involves inversion of certain bits while port 37AH does not when used for writing a byte of data to the parallel port to control the logic state of each pin.

For faster access and greater flexibility, software should directly access these parallel ports through the three addresses associated with it. As used in the PC, parallel port 1 has address 378H for writing a byte of data to the parallel port pins, address 37AH enables writing control signals, and address 379H gives input signals from a device. Therefore, to output a byte of data, the program writes a character to address 378H, and input may be done at address 379H and 37AH.

Features such as OUT and INP() instructions in BASIC, P o r t [ ] arrays in Turbo Pascal, and i n p o r t b () and out p o r t b () functions in C enable the direct manipulation of the parallel port through these addresses.

For example, to write the hex value 27 to port 378H in Turbo Pascal, we can write:

```
Port [$378] := $27;
```

Thus, to read data, we use the statement:

```
SomeVariable = Port [$379];
```

where SomeVariable is a variable capable of storing a byte of data

## THE NYBBLE INDICATOR APPROACH

Assume for a moment that we are exchanging data between two computers using the parallel port. The receiving computer needs to read data from the parallel port, but can only do so five bits at a time. This is an inconvenient quantity, so we will just use four of these bits for a nybble of data.

Typically, when you manipulate nybbles with software, you designate one pin of the port to act as a nybble indicator. The sending program is responsible for splitting the data into two nybbles and transfers it via four of the five input pins on the parallel interface.

One example of such a software transfer may be seen in the code segment in Listing 1. If the sending party is a piece of software, then the receiving program indicates with the NYB\_IND pin which nybble is to be sent.

## IN HARDWARE

If instead hardware is used to transmit data, a quad 2-to-1 multiplexer solves the problem. Figure 1 shows the schematic connecting the device to a standard parallel interface.

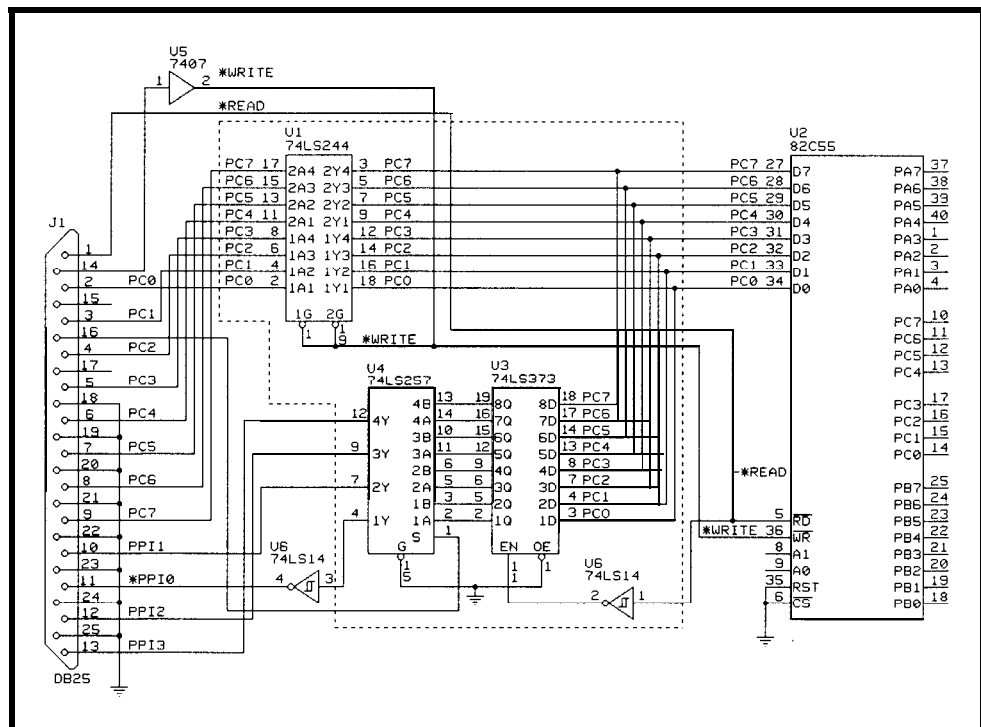


Figure 1-In the Nybble Indicator approach, a bit is used to designate whether the high or the low nybble is being transferred.



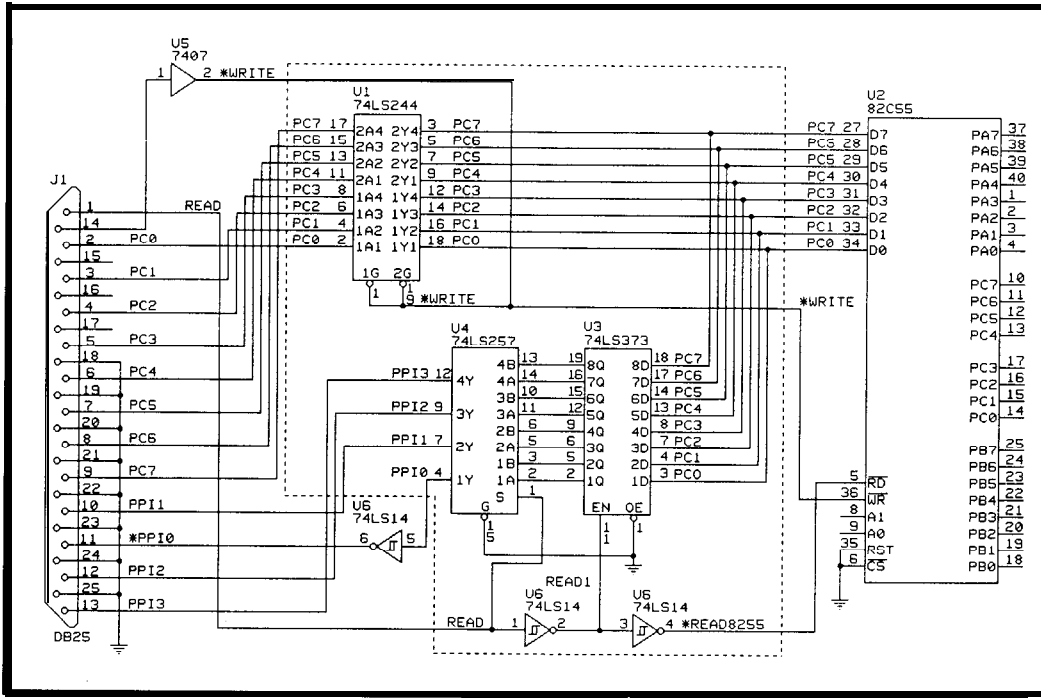


Figure 2—The Read Edge approach differs from the previous method in its use of the RD line.

In our example, the nybble indicator uses pin 16, which corresponds to bit 2 on port address 37AH.

Looking at the original DDT-5 1, the data bus consists of pins 2-9 on the parallel-port connector—it became bidirectional through hardware modifications. If the port is left unmodified, we have to add a 74LS244 (U2), which is used to isolate the data bus of the 8255 from the parallel port. U2 is required as the DO-D7 pins on the standard parallel port are constantly outputting data to the data bus, which causes problems during an input operation. Pins 11, 10, 12, and 13 are now used for input to the PC, which correspond to bits 7, 6, 5, and 4 on port address 379H.

The dotted area in Figure 1 shows the change to the original circuit. In this approach, the sequence of reading a byte begins by setting the READ line high. To initiate a read operation, the line changes to low., The software then pulls the NYB\_IND (active) high and reads the first nybble from the 74LS257. It then saves the nybble and pulls the NYB\_IND low to read in the next nybble. The read operation ends by setting the NYB\_IND and READ to high again. The -A/B pin of the 74LS257 is connected to the NYB\_IND pin on the parallel port.

## THE READ EDGE APPROACH

Now, let's take a look at what we did. For reasons that will be apparent later, we will call this the *Read Edge method*. Figure 2 shows the schematic with the new circuitry. We use a pair of Schmitt-triggered inverters (U7, U8) to clean up the READ line. The difference between the methods hinges primarily in the use of the READ signal. Here, together with the additional ICs, as shown in the figure, it is also used as the nybble indicator.

When the software needs to read a byte of data, it pulls the READ line

low. This causes U5, a quad 2-to-1 74LS257, to output the low byte to the input portion of the parallel port. (Again, we are using bits 7, 6, 5, 4 of port 379H, which correspond to pins 11, 10, 12, 13 on the parallel port.) The signal causes U4, a 74LS373 latch, to pass data through to U5.

On receiving the READ8255 signal, the 8255 places the data on the bus, which connects to U4. Since U4 is a transparent latch, data passes through it when LE, which is connected to the READ1 signal, is high. This allows the data to appear at the input of U5, which selects the

lower four bits of the byte and outputs this nybble. The READ signal, which acts like the NYB\_IND, goes to the -A/B input of the multiplexer to determine which nybble to place on the input pins. In this case, it is low for the lower nybble and high for the higher nybble. These nybbles are stored in the PC.

The software then pulls the READ line high. The LE on U4 then goes low as the first Schmitt-triggered inverter (U7) reverses the logic level of the signal. The falling edge of this signal latches the output from the 8255.

Listing 2—The READ8255 routine in the DDT-51 software has to be modified to work with the Read Edge hardware.

```

Function 8255 : Byte;
Var HighByte, LowByte: Byte;
Beq in

Port [CTLPort1 := CstbCR;           DDT-51 port values)
Port [CTLPort1 := Cread;

Port [CTLPort1 := CstbRD;           Pull RD low}
LowByte := (Port [InpPort] And $F0 SHR 4

Port [CTLPort1 := Cnull;           Pull RD high}
HighByte := Port [InpPort] And $F0

Read8255 := HighByte Or LowByte;    Combine nybbles}

End:

```

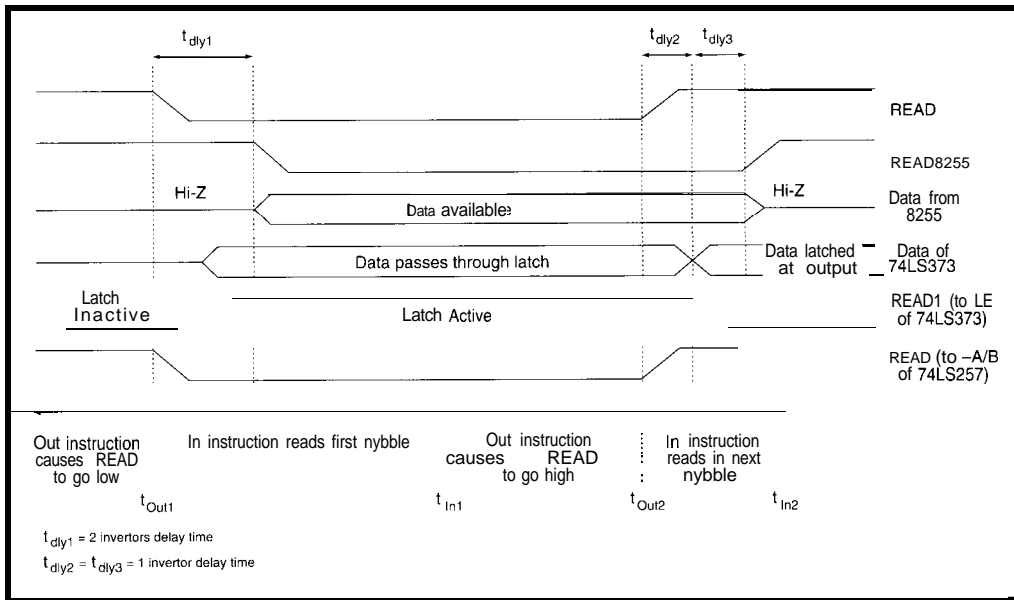


Figure 3-A timing diagram helps in analyzing the Read Edge approach for any possible delay problems.

After the rising edge of the READ8255 signal, the 8255 releases the data bus to a high-impedance state at the same time or before U4 is able to latch the data. The second inverter (U8) delays the READ8255 signal so that U4 (the 74LS373) has time to latch the data.

Latching is done before the READ1 signal propagates through the second 74LS14, which pulls the READS255 line at the 8255 high. When this happens, the output from the 8255 floats. Looking now at U5, the 74LS257 multiplexer, the high READ signal outputs the high byte of the data. The software reads the nybble and then combines them to form the final byte. Figure 3 shows the timing diagram for the read operation.

### TIMING ANALYSIS

Let us analyze the timing of the signals to see the potential and limits of this approach. Look at time  $t_{Out1}$  in Figure 3. After the CPU brings the READ line low, an I N instruction reads in the first nybble. Because of the timing delay of  $t_{dly1}$ , the first nybble must be ready to be read by  $t_{In1}$ . However, the read for the second nybble always finds data ready at  $t_{In2}$ .

Now, if the CPU wants to do a write to the 8255 instead, the time  $t_{dly2}$  and  $t_{dly3}$  causes the 8255 to change from a READ state that is much slower.

The inverter delays are on the order of 10 ns using standard LS logic. An I N instruction takes hundreds of nanoseconds to execute for a PC. Some of you may frown at using delays to achieve our objectives, but we have demonstrated that it does work. Incorporating the circuitry into a PLD may lessen some of the layout problems.

### THE SOFTWARE

So far, we have been talking about the hardware that converts bytes into nybbles for the PC parallel port. However, the approach of reading such a byte of data also requires the use of software to generate the necessary read signal and assemble the two nybbles into a byte.

In the DDT-5 1 project, we modified the READ8255 function to achieve this. Listing 2 shows the code samples written in Turbo Pascal. Note that there are two variables HighByte and LowByte, both of the data type BYTE.

When a statement calls the function READ8255, the READ line on the parallel port goes active (low). This causes the hardware to respond with the lower nybble at the input pins. The software then reads the nybble from the input pin through the address 379H. Since each read operation at address 379H yields a byte of data, the software ignores any unused bits in the byte of data and zeros them before storing them into the variables. From Table 1, we see that the four input pins lie in the upper four bits of the byte.

Depending on the state of the READ signal, the software either shifts the nybble four bits to the right or no shifting is performed at all.

As you can see in Listing 2, the first nybble read represents the low

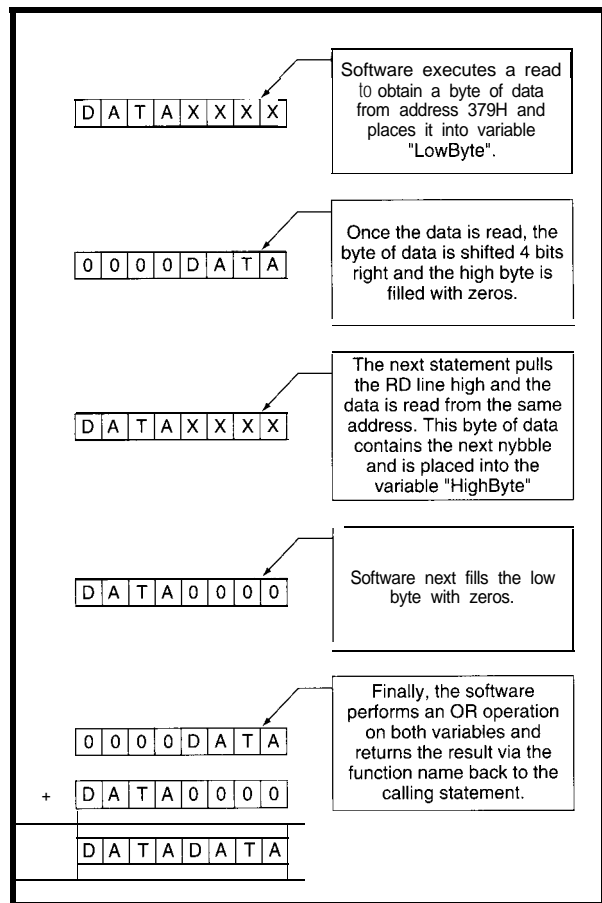


Figure 4—The nybble manipulations needed to recover a byte of data when only four bits of the data are transferred at a time are readily coded in most languages.

byte. The software shifts this nybble four bits to the right and fills the upper nybble with zeros. It then pulls the READ line high to read the next nybble. The second nybble, which represents the high byte, need not be shifted at all since it is already in its correct position. The lower nybble is then filled with zeros.

Once the software stores the two nybbles in the variables `HighByte` and `LowByte`, they are ORed together. The last statement in the function then places the result into the function name, which returns the value to the calling statement. Figure 4 illustrates the process of the software performing

the read and assembling the two nybbles.

## ASSEMBLY LANGUAGE

Writing the code using a high-level language to read the nybbles and assembling it to form the final byte of data may be too slow for certain applications. To solve this, you can use an optimizing compiler to produce a faster routine or simply write the `READ8255` function in assembly language to begin with.


We have converted the `READ8255` function to C and used Borland C++ 3.1 to compile the program. By turning on the speed-optimization option, the

compiler was able to generate a fast routine for the read operation that was almost as fast as one we hand-optimized in assembler.

Turbo Pascal, however, produces a final assembly listing with some slight overhead in the code. The overhead may be avoided by simply writing the assembly version. We provide a sample of the assembly version of our `READ8255` in Listing 3.

## CONCLUSIONS

Although the Read Edge approach allows the software to read a byte of data using a standard parallel port with a single READ signal, it cannot be used on two PCs without this hardware between them.

But, this method is one of the many ways to read data using a parallel port. Its main advantage lies in the use of the READ signal to differentiate the nybbles in the data byte. This saves one pin on the parallel port which may be used for other purposes such as a control line for the attached device. Furthermore, this technique reduces the complexity of the software as less coding is required. 

*Tracey Lee specializes in micro-systems in the Department of Electronics, Computer, and Communications Engineering at the Singapore Polytechnic. He worked for IBM Singapore for 8 years and has a B.Eng. from Singapore University and a M.Eng. from Nanyang Technology University. He may be reached at [spec02@solomon.technet.sg](mailto:spec02@solomon.technet.sg).*

*Kok-Leong Ong recently finished his studies at the Singapore Polytechnic where, together with two other students, he worked on a final year project entitled "8051 Emulator."*

## REFERENCE

Ciarcia, Steve. "Why Microcontrollers?" *BYTE*, Aug.-Sept., 1988.

## I R S

- 410 Very Useful
- 411 Moderately Useful
- 412 Not Useful

**Listing 3**—The routines to read and write data to the standard PC parallel port are generic enough to be useful for most applications. However, the `ReadPort` routine is written specifically for the DDT-51 project. Modify the code to suit your application.

```

DOSSeg
.model small
.stack 100h
.data

OutputPort equ 378h
InputPort  equ 379h
ControlPort equ 37Ah

.code

public ReadPort

ReadPort proc

    push dx
    push cx

    mov dx, ControlPort
    mov ax, 29h
    out dx, ax           ; assert RD line low
    mov dx, InputPort
    in  al, dx           ; read the nybble
    and al, 0F0h         ; zero the lower nybble
    shr al, 1           ; faster to do 4 * 1
    shr al, 1           ; bit shifts
    shr al, 1
    shr al, 1
    mov ch, al          ; save nybble

    mov dx, ControlPort
    mov ax, 08h
    out dx, ax           ; pull read line high
    mov dx, InputPort
    in  al, dx           ; read upper nybble
    and al, 0F0h         ; zero lower nybble
    or  al, ch           ; merge nybbles into bytes

    pop cx
    pop dx
    ret
endp
end

```

# Battery-operated Power Supplies

## FEATURE ARTICLE

David Prutchi

## Selecting the Right Battery and Supply for Your Application

Matching batteries to design specifications is critical. To help developers with this task, David offers an overview of batteries—primary, secondary, backup, and their use in driving power supplies.



When designing self-contained, portable, or isolated instruments, one of the main concerns is often the power source for the system. These applications usually can't rely on a line-operated power supply and thus a battery-operated source becomes necessary.

A number of battery, charger, and DC/DC technologies are currently available. However, careful consideration of many factors is imperative in selecting the right kind of battery for a given application.

On one hand, the electrical attributes of the battery constitute the design constraints of the power supply, along with the load characteristics, power demand, and required operation time. On the other hand, physical and safety conditions restrict the number of technologies that conform to the electrical specifications. Finally, battery price and maintenance costs narrow the remaining alternatives.

In addition, the mission objectives for the instrument must be carefully analyzed to determine whether a nonrechargeable battery (also called a

primary battery) or a rechargeable battery (secondary battery) should be used.

Once a battery matches the required specifications and is selected, an appropriate DC/DC converter must be designed to supply the various subsystems of the instrument with the voltage levels each requires. If rechargeable batteries are used, charging and maintenance circuitry must be designed.

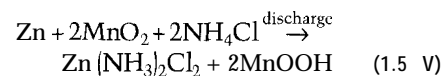
This article discusses proper battery system design for self-contained microcomputer applications. Battery selection considerations as well as power-conversion design is also addressed.

### PRIMARY BATTERIES

Primary batteries convert chemical energy irreversibly into electrical energy. As such, they are not meant to be recharged. The main advantages of these batteries are their high-energy density, good shelf life, wide availability, and maintenance-free operation [1].

Standard zinc-carbon cells have a thin, solid zinc (Zn) casing for an anode. This is surrounded by a moist cake of manganese dioxide powder ( $MnO_2$ ) and an electrolyte solution of granulated carbon and zinc chloride ( $ZnCl_2$ ) blended together with ammonia ( $NH_3$ ). A carbon-rod electrode is introduced into the solution.

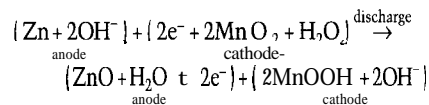
Energy for the discharge current is obtained by turning zinc into zinc diammine chloride ( $Zn(NH_3)_2Cl_2$ ) and by reducing the manganese dioxide to manganese oxyhydroxide ( $MnOOH$ ) through the reaction:



Standard zinc-carbon cells have not changed all that much since they were introduced over 100 years ago by Leclanché. Their main advantage is that they are the cheapest batteries that can be bought. Their main disadvantage is that their voltage drops and their internal impedance increases steadily as the battery is used. Moreover, their capacity is severely reduced when used at high currents.

**Heavy-duty zinc-carbon cells**—are improved Leclanche cells which incorporate higher zinc-chloride content and modified design to cope with higher gas generation during operation. Although their energy density is only slightly higher than that of standard zinc-carbon cells, their capacity is not as strongly affected by high-current operation. In addition, the tolerance of heavy-duty cells to temperature is somewhat better than that of standard cells.

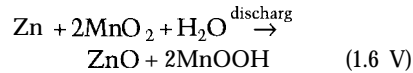
**Alkaline (Zn-MnO<sub>2</sub>) cells** are the most widely used household batteries today. Inside them, zinc is used as the anode material and manganese dioxide as the cathode. Electrochemical activity takes place through a potassium hydroxide electrolyte. The energy-producing reaction involves oxidizing the zinc anode to form zinc oxide (ZnO), while reducing the manganese dioxide to manganese oxyhydroxide. This reaction, which supplies electrons (e<sup>-</sup>) for the discharge current, can be expressed as:



which is simplified to:

Characteristic	Zinc-Carbon Standard	Zinc-Carbon Heavy Duty	Alkaline	Mercury	Lithium (various)
Capacity (Ah max., largest available)	3	5	20	1	8
Energy Density (Wh per inch <sup>3</sup> max.)	2	2.5	3.5	7	8
Temperature Range (0°C @ 85% capacity)	10-55	5-65	2-70	3-75	-30-100
Cell Voltage (V, full/empty)	1.5/0.45	1.5/0.7	1.5/1	1.4/1.3	413.8

Table I—Typical primary batteries vary in their charge capacity, current-supplying capability, and cell voltages. "Capacity" refers to the largest standard commercial battery that is widely available.



Because of their low internal resistance, low currents can be drawn from alkaline cells for very long periods of time, and they have a moderate energy density. As with zinc and carbon cells, their main disadvantage is a sloping discharge curve. In addition, the largest serial-connection battery packs can achieve a maximum current of about 1 A @ 4 Ah using D-size cells.

In general, alkaline batteries are very inexpensive, widely available, and have relatively good shelf life. These characteristics make them the favorite choice for household battery-powered equipment, but their degraded performance at low and high temperatures should be of real concern to designers of equipment exposed to harsh environments.

**Lithium cells** are one of the newest commercial primary cells exhibiting very high energy density. Although there are various lithium-based chemistries available, the common advantages to these cells are

their wide temperature range of operation, extraordinary shelf life, and very good discharge voltage regulation. One large disadvantage of lithium cells is that they still present a safety hazard because of their extremely high energy density and the toxicity of their contents [2].

One of the most common lithium-cell chemistries is based on lithium manganese dioxide (LiMnO<sub>2</sub>), which comes in a variety of packages. Because of their long shelf life, these batteries are commonly used as memory backup supplies. However, high-power LiMnO<sub>2</sub> cells are used today as the main power source for intelligent cameras, flashlights, data-acquisition packages, test and measuring instruments, and so on.

Similar to alkaline batteries, the energy-producing reaction involves metallic lithium (Li) and manganese dioxide. The exact reaction path is still unknown, but it has been suggested as:

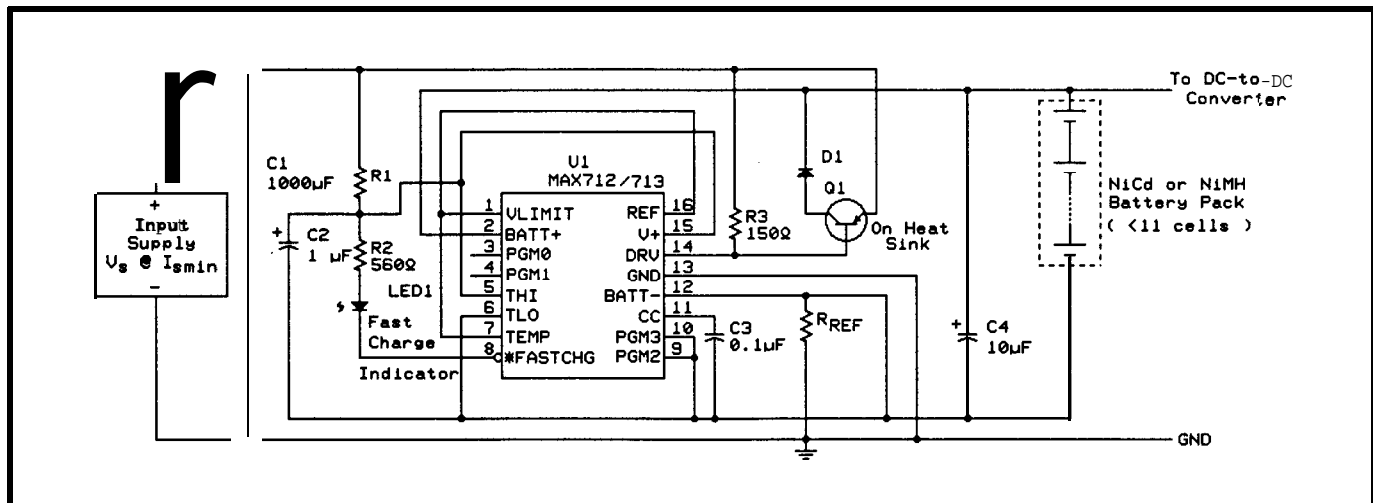
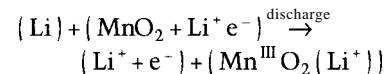


Figure 1—Maxim's new MAX712/MAX713 NiCd/NiMH battery fast-charge controllers can be the basis of a low-cost, low-parts-count battery charger circuit. See the article for guidelines on how to pick values for the unspecified components.

where  $Mn^{III}O_2(Li^+)$  represents the introduction of a lithium ion ( $Li^+$ ) into the manganese dioxide crystal lattice. This last equation is simplified for the overall cell reaction to:



Another kind of lithium cell in use today is based on lithium thionyl chloride ( $LiSOCl_2$ ). This cell is replacing older alkaline and lithium chemistries in military instruments.

**Other primary batteries-with** different chemistries are commercially available. However, unless a critical design constraint demands their use, the best idea is to select the most widely available batteries and to avoid hard-to-get types. Two battery chemistries that should nevertheless be mentioned because of their extensive use in miniaturized instruments are:

- silver oxide cells-These batteries are typically used in watches and cameras. They have very flat discharge characteristics, but have a shorter shelf life than lithium batteries.
- zinc-air batteries-These generate electricity through a chemical reaction which uses the oxygen in the air. They have extremely high power density and, for this reason, have started to gain ground as a replacement for mercury batteries in hearing aids, medical telemetry equipment, and pagers.

Table 1 offers a summary of the primary batteries discussed.

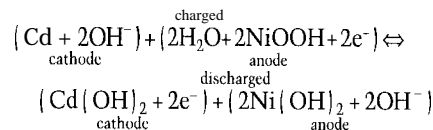
## SECONDARY BATTERIES

Secondary batteries are designed around chemical reactions with electrochemical reversibility, which makes them rechargeable. These batteries offer cost savings over

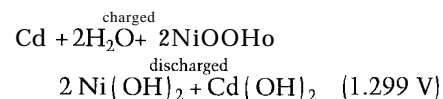
the life of the product they empower and provide an effective method of storing energy for cordless operation [3].

**Nickel-cadmium (NiCd)** cells take advantage of a reversible electrochemical reaction in which the active electrode materials change in oxidation state without any physical deterioration. This property gives NiCd cells a relatively long life.

NiCd cells use a nickel oxyhydroxide ( $NiOOH$ ) anode, a cadmium (Cd) metal cathode, and a potassium hydroxide (KOH) electrolyte. During the charge cycle, the anode is reduced to nickel hydroxide ( $Ni(OH)_2$ ) in an endothermic (heat-absorbing) reaction. During discharge, the anode returns to its primary state, while the cadmium metal is oxidized to cadmium hydroxide ( $Cd(OH)_2$ ), releasing electrons to the load. These reactions can be expressed as:



which is simplified to:



Number of Cells (n)	PGMO Connection (Pin 3)	PGM1 Connection (Pin 4)
1	V+ (pin 15)	V+ (pin 15)
2	V+ (pin 15)	n/c
3	V+ (pin 15)	REF (pin 16)
4	V+ (pin 15)	BATT- (pin 12)
5	n/c	V+ (pin 15)
6	n/c	n/c
7	n/c	REF (pin 16)
8	n/c	BATT- (pin 12)
9	REF (pin 16)	V+ (pin 15)
10	REF (pin 16)	n/c

Table 2-"Programming" the MAX712/MAX713 to charge an n-celled NiCd/NiMH battery is done by connecting two of its input pins in different combinations.

The endothermic characteristic of the charging reaction offsets the heat produced by the inefficiency of the charging process, resulting in an almost constant battery temperature during the charge cycle. However, on completion of this cycle, cell overcharge leads to a sharp increase in battery temperature. A sensor monitoring this temperature change may use this as a signal to terminate battery charging. This temperature shift is often used in the control of fast charging. High charging current must be reduced immediately at the moment of temperature rise to prevent the development of excessive gas pressure.

When NiCd cells move into the overcharge region, the voltage peaks and then begins a clear-cut decline. Detection of the resulting negative slope of the voltage curve can also be used to terminate charging. Depending on the design and manufacture process, modern NiCd cells can be

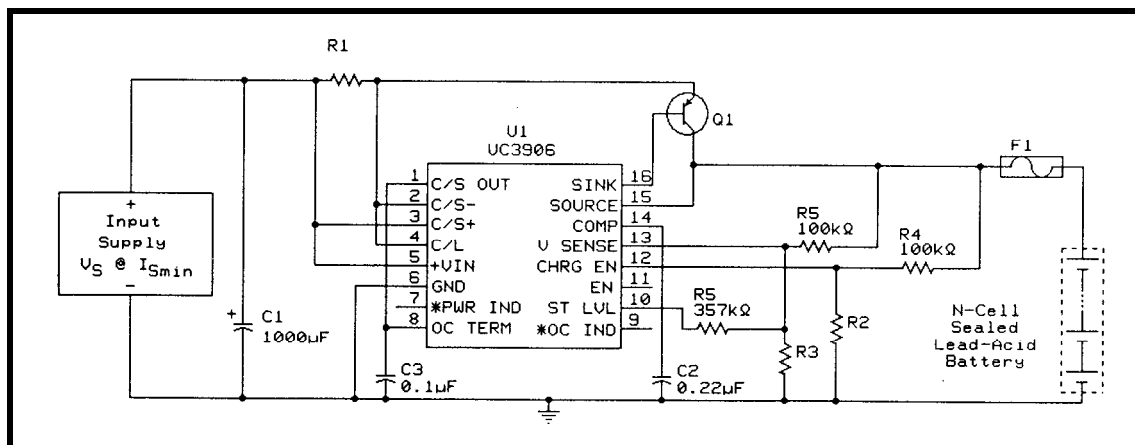


Figure 2-A UC3906 is at the core of the design of a sealed-lead-acid battery charger which implements an optimal charging and maintenance algorithm. Again, see the article for component-value selection guidelines.



	NiCd	NiMH	Sealed Lead-Acid
Capacity (Ah max., largest available)	5	8	25
Energy Density (Wh/in. <sup>3</sup> )	1.2	1.8	1
Temperature Range ("C, @ 85% capacity)	-15 to 40	-20 to 45	-50 to 50
Cell Voltage (V, full/empty)	1.35/1.1	1.41/1.1	2.4/1.6
Self-Discharge Rate (% per month)	25	30	6
Number of Full Cycles	800	500	200 to 2000

Table 3—While sealed-lead-acid batteries might appear to be the device of choice when selecting a secondary battery, they are much larger and heavier than NiCd and NiMH batteries.

fast-charged at rates from  $\frac{C}{3}$  (where C is the rated battery capacity) up to 4C. After this point, a  $\frac{C}{16}$  trickle-charging rate should be used to maintain full battery capacity during idling conditions.

Older NiCd cells, on the other hand, do not accept high charging rates above  $\frac{C}{10}$ . It should be noted that fast-charging rate selection is critical because rates higher than those specified for a cell cause electrolyte seal leakage and gas-bubble formation.

Figure 1 presents the circuit of a NiCd battery-pack charger. The heart of the circuit is Maxim's new NiCd battery fast-charge controller IC, the MAX713. It is capable of charging up to 16 series cells and uses negative-voltage-slope detection and an internal timer to switch automatically from fast to trickle charging.

Under a different circuit configuration, the MAX713 can also use temperature sensing to terminate fast charging. In addition, the load can be powered while charging the batteries without affecting the controller's response. Of course, the input supply should be capable of supplying enough current for both battery charging and powering the load.

For a simple NiCd charger with a fast-charging current ( $I_{fast}$  [A]) of less than 0.6 A and a battery of less than 11 cells (< 15 V), unregulated DC input supply can be selected for the circuit of Figure 1 using the following:

$$V_s = V_{battery} + 3 [V] @ I_{s, min.} = C [A] I_{fast} = \frac{C}{2} [A] I_{fast}$$

$$R1 = \frac{V_s - 5}{5} [K\Omega]$$

$$R_{sense} = \frac{0.25}{I_{fast}} [\Omega] @ 0.5 I_{fast} [W]$$

Note that DI's minimum current handling equals  $I_{s, min}$  (a 1N4001 is usually suitable) and Q 1's minimum power dissipation equals  $(V_s - V_{discharged, battery}) I_{fast} [W]$ .

Connecting lines PGM2 and PGM3 to BATT program the MAX712 to timeout in a period of 264 minutes and to use a fast-charge rate of  $\frac{C}{2}$ . In addition, the MAX712 must be told how many cells there are in the battery pack ( $V_{battery} \approx 1.3n [V]$  where n is the number of cells). This is accomplished by strapping lines PGMO and PGM1 according to Table 2. For design specifics of the MAX712 and MAX713, see [4].

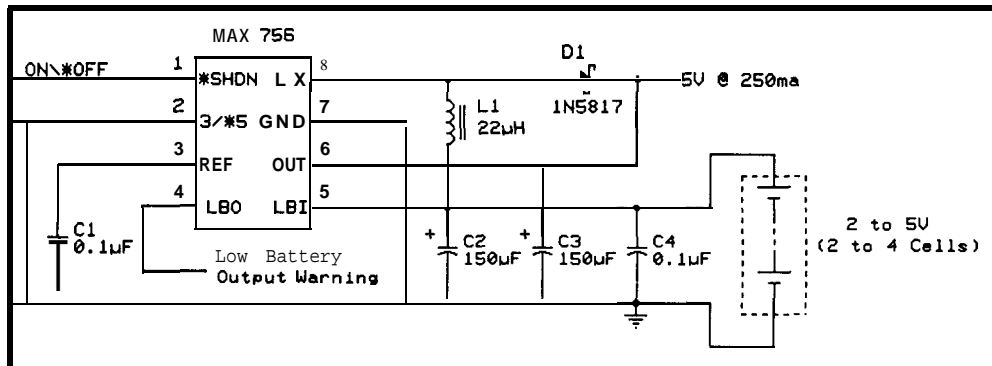


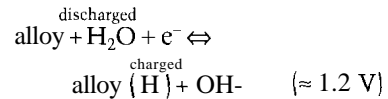
Figure 3—It's possible to achieve 87% efficiency with this MAX722-based DC-DC converter. In this case, it runs on 2 to 4 cells and provides regulated +5V.

The discharge of NiCd batteries should be monitored to maximize their performance and useful life. When the load is connected to a NiCd battery, the cell voltage should not be allowed to drop under 1.1 V as this may result in permanent plate and/or electrolyte damage. Monitoring can be accomplished through a simple comparator circuit that disconnects the load at the right time.

Further, the battery should be allowed to completely charge and discharge (down to 1.1 V per cell) at least 60% of the time or the battery's

capacity will fall. For an in-depth review of how to charge and use NiCd batteries, refer to [5,6].

**Nickel-metal hydride (NiMH)** cells are starting to appear in the market. These batteries have similar characteristics to those of high-performance NiCd types, but have as much as 30% higher energy density (almost twice the energy density of standard NiCd cells). Their operational principle lies on the ability of certain metallic alloys to reversibly absorb hydrogen atoms into the metallic structure of the alloy:



NiMH batteries are similar in construction to NiCd batteries, except that proprietary alloy formulations are used to form the metal hydride cathode instead of cadmium.

NiMH charging methods are similar to those used on NiCd cells.

However, the selection of charging control and termination parameters must take into account that NiMH batteries heat up during charging. Although they exhibit a sharp temperature increase on overcharge, the negative slope to the voltage curve at the end of charging is much less marked than that of NiCd.

The circuit charging NiCd batteries can also be used to charge NiMH batteries by replacing the MAX713 with a MAX712, which takes into consideration these differences.

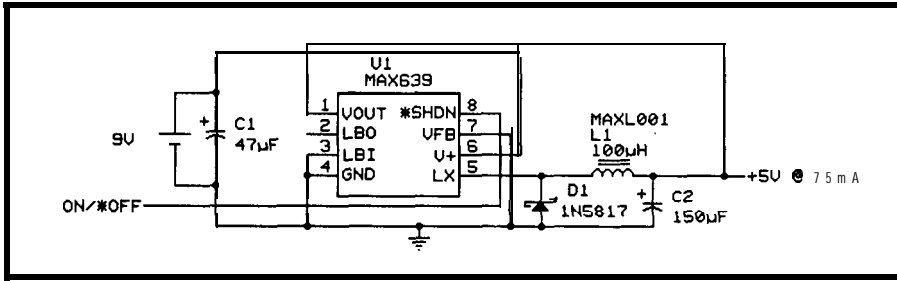
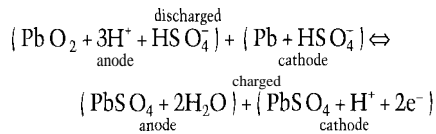


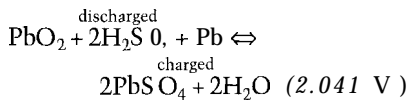
Figure 4-A +5-V output, 94% efficiency DC-DC converter exploits the full capacity of a 9-V alkaline battery.

Sealed lead-acid (SLA) batteries—share the same chemistry of automotive lead-acid batteries. During the discharge cycle, a lead dioxide (PbO<sub>2</sub>) anode and a metallic lead (Pb) cathode react with the sulfuric acid electrolyte (H<sub>2</sub>SO<sub>4</sub>) to form lead sulfate (PbSO<sub>4</sub>) and water. The electrons are given to the discharge circuit.

During the recharge cycle, the same reactions are reversed. These reactions can be expressed as:



which can be simplified for the overall cell reaction as:



SLA batteries differ from “flooded” lead-acid batteries in that a minimum amount of electrolyte is used. Their maintenance-free operation is the result of a gas-recombination technique in which the oxygen and hydrogen generated within the cell are recombined to maintain the amount of water needed for operation.

SLA cells have a very low internal resistance, which allows them to source a high current. Their primary advantage is their cost and the high number of cycles they can withstand. They have good storage-capacity retention and discharge-voltage regulation. In addition, the temperature range for their operation exceeds that of most other secondary batteries. The main disadvantage of SLA cells is their relatively low energy density, which makes SLA batteries much

heavier and voluminous than other secondary batteries of the same capacity.

During the charge cycle, a SLA battery is typically brought up to overcharge at about 2.4 V per cell under a bulk charging rate of  $\frac{C}{4}$ . When the final charging voltage is reached, charging current through the battery is slowly dropped to a trickle rate of  $\frac{C}{10}$  to maintain a float voltage of 2.25-2.3 V per cell. Without this trickle charge, stored or idling SLA cells lose a considerable amount of their capacity.

Unitrode’s UC3906 sealed-lead-acid battery-charger chip implements an optimal charging algorithm for these batteries. The circuit in Figure 2 is a typical application for the UC3906. The circuit is supplied with unregulated DC at:

$$V_s = n ( 2.4 ) + 3 \text{ V} @ I_{s \text{ min.}} = \frac{C}{6}$$

where  $n$  is the number of cells in the battery and  $C$  is the nominal 20-h rate capacity of the battery. A simplified set of equations to select component values for the circuit is:

$$\begin{aligned} R1 &= \frac{4}{C} [\Omega] \\ R2 &= \frac{230}{1.3n - 2.3} [\text{K}\Omega] \\ R3 &= \frac{100}{n - 1} [\text{K}\Omega] \end{aligned}$$

$$\text{Fuse rating} = \frac{C}{3} [\text{A}] \text{ slow-blow}$$

Transistor Q1 should be chosen to meet charging current and voltage requirements, taking into consideration that the output drive current of the UC3906 is limited to 25 mA. For detailed design information, refer to [7]. Specific applications using the UC3906 can be found in [8,9].

Other secondary batteries are used to power self-contained instru-

# FREE Data Acquisition Catalog



# 1995

PC and VME data

acquisition catalog

from the inventors of

plug-in data acquisition.

Featuring new low-cost

A/D boards optimized

for Windows,

DSP Data Acquisition,

and the latest

Windows software.

Plus, informative

technical tips and

application notes.

Call for your free copy

**1 - 800 - 648 - 6589**

## ADAC

American Data Acquisition Corporation  
70 Tower Office Park, Woburn, MA 01801  
phone 617-935-3200 fax 617-938-6553

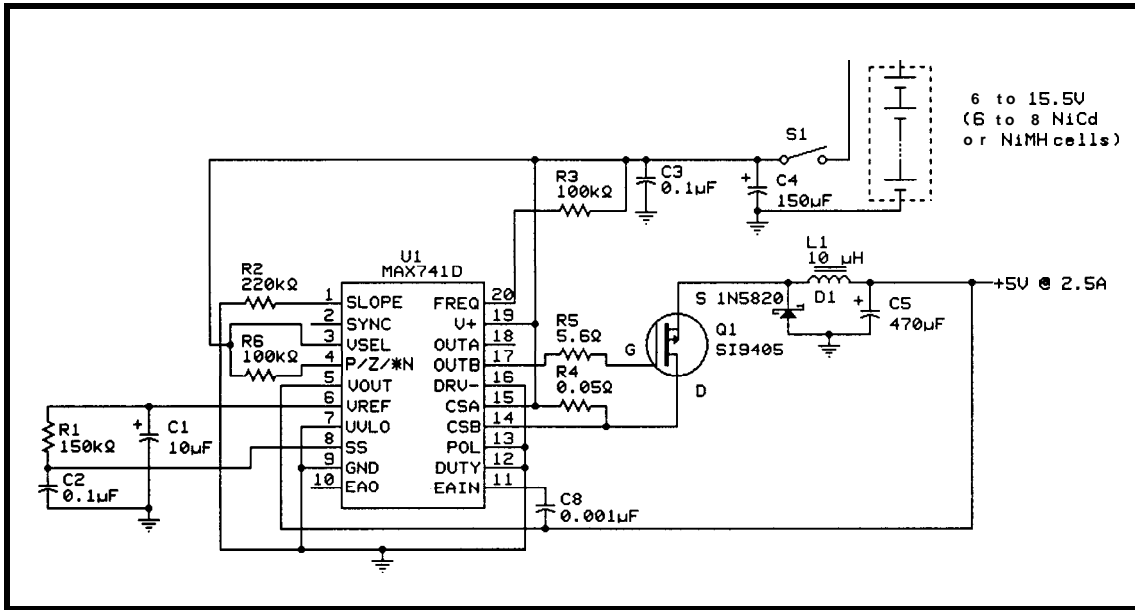


Figure 5-A medium power (+5V @ 2.5 A max.) DC-DC converter can be used to power a notebook-type microcomputer using a rechargeable NiCd or NiMH battery pack.

about 5 years. Once they are loaded with a liquid potassium hydroxide electrolyte, they can be charged up to 200 times within a two-year life period. Although they are expensive, require considerable maintenance, and have a relatively limited life, they are the battery of choice for applications in which weight and volume are the main concern because of their attributes.

ments. However, these are not currently available for consumer applications. Their cost and lack of availability should deter the experimenter from their use. Nevertheless, two other technologies are worth mentioning:

\* *Silver-Zinc cells* are usually used for military and aerospace applications and are not common in commercial markets. These cells are kept without electrolyte until the day they are put to use. Their dry shelf life is

They provide large energy density, excellent capacity retention, and almost-flat discharge characteristics. *Lithium-ion batteries* were only recently introduced to the market.

**Cimetrics**  
TECHNOLOGY

*Linking Microcontrollers.  
Breaking Boundaries.*

**The 9-Bit Solution**

The Cimetrics Technology 9-Bit Solution is a complete microcontroller network (μLAN) that supports the 8051, 68HC11, 80C186EB/EC, and many other popular processors. The 9-Bit Solution takes full advantage of microprocessor modes built in to microcontrollers. The 9-Bit Solution allows simple and inexpensive development of master/slave multidrop embedded controller networks.

- 8051, 68HC11, 80C186EB/EC compatible
- A full range of other processors supported
- Up to 250 nodes
- 16 Bit CRC error checking with sequence numbers
- Complete source code included

**Link Your Product  
With A Cimetrics  
μLAN Today!**

**617.350.7550**

55 Temple Place • Boston, MA 02111-1300  
Ph 617.350.7550 • Fx 617.350.7552

**3½-DIGIT LCD PANEL METER**  
-Available now at an unheard of price of **\$15** plus s & h  
**New! Not surplus!**

**Specifications:**

- Maximum input: f199.9 mV  
additional ranges provided through external resistor dividers
- Display: 3½digit LCD, 0.5 in. figure height, jumper-selectable decimal point
- Conversion: Dual slope conversion, 2-3 readings per sec.
- Input Impedance: >100M ohm
- Power: 9-1 2 VDC @ 1 mA DC

**Circuit Cellar, Inc.**  
4 Park Street, Suite 12, Vernon, CT 06066  
Tel: (203) 875-2751 Fax: (203) 872-2204

These rechargeable batteries, with 3.6 V per cell, have similar characteristics to those of primary lithium batteries, and feature about twice the energy density of NiCd cells.

These batteries are worthy of special note particularly because the first is often encountered in government-issue equipment and is of interest to the surplus hunter, and the second has great potential.

A summary of secondary battery technology is in Table 3.

### DESIGN OF BATTERY-OPERATED POWER SUPPLIES

Many battery-powered instruments are designed to operate directly from the battery without further power transformation or regulation. This design approach offers a low parts count and a very high power efficiency, but unfortunately is seldom a viable option. Power conversion or regulation is generally required.

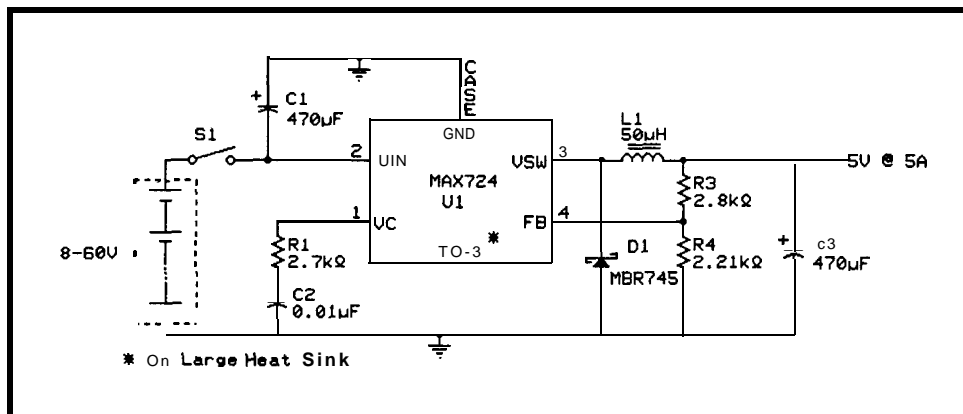
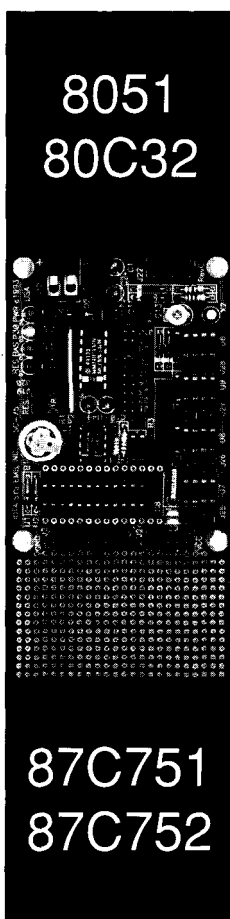


Figure 6-A high-power (+5 V @ 5 A max.) DC-DC converter uses high-capacity SLA batteries to provide input power.

Whenever step-down regulation is required, three-terminal linear regulators may be used to keep the circuit cheap and small since they are widely available, simple to use, and don't require support components, except for decoupling capacitors. However, linear regulators are seldom used in power-conscious designs because they are less efficient than modern switching regulators. In addition, linear regulators cannot step up or invert battery voltages.

One of the simplest +5-V-output DC-DC converters for battery-powered applications is depicted in Figure 3. Up to 87% efficiency can be obtained by using this circuit. Maxim's MAX756 delivers +5 V at 250 mA from 2 to 4 cells and even includes an on-chip, MOS, digitally controlled power switch and a power-fail signal to warn the microprocessor when the output falls out of regulation.

The 9-V "1604" batteries are appealing to hand-held instrument



8051  
80C32

Use one of our embedded controllers to save time and money. They are ideal for developing products, test fixtures and prototypes.

We offer a complete line of controller boards and software tools for the 8051 and 87C751 families of microcontrollers. Complete packages are available to help you develop your projects.

#### Features:

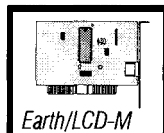
- Breadboard area
- Flexible I/O arrangement
- Powerful controller BASIC for the 87C752 or 80C32

Ph: (702) 83 1-6302  
Fax: (702) 83 1-4629

87C751  
87C752

Iota Systems, Inc.  
POB 8987 • Incline Village, NV  
89452-8987

#### L.C.D. VGA CONTROLLER



Mono L.C.D.s Supported:  
640 x 480  
640 x 200  
320 x 200  
480 x 128

ISA 8 BIT  
PC. COMPATIBLE  
WINDOWS DRIVERS  
HARDWARE PANNING  
RUNS VGA PROGRAMS  
ON LOW RES LCDs  
INCLUDES 10% OFF ANY  
LCD PANEL DISCOUNT  
LOW COST - \$149

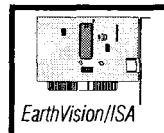
#### L.C.D. VGA SCREENS



LOW COST DISCOUNTED 30%-60%

MONO REFLECTIVE  
VGA 9.4" [IN A CASE!] \$40  
VGA 7" \$50  
MONO BACKLIT  
VGA 9.4" 250 MS \$85  
VGA 9.4" 150 MS \$99  
9" CGA 640 X 200 \$19  
COLOR BACKLIT  
VGA 9.4" SNGL SCAN \$99  
VGA 9.4" DUAL SCAN CALL  
VGA TFT 9.4" CALL

#### COLOR L.C.D CONTROLLER



Color L.C.D.s Supported:  
Single Scan  
Dual Scan  
TFT

ISA 16 BIT VGA  
WINDOWS ACCELERATOR  
512K OR 1M RAM  
CHIPS & TECH 65545  
FASTEST LCD CHIP  
LOW COST \$299

#### 486 PC. IN A KEYBOARD



33 Mhz 486SX  
1-32MB Ram  
SuperVGA  
2 serial/ 1 Par.  
3.5" Floppy  
3.5" H.D.

101 KEYBOARD FOOTPRINT  
FULLY PC AT COMPATIBLE  
ONE 16 BIT ISA SLOT  
512K FLASH DRIVE AVAILABLE  
ETHERNET DISKLESS FROM  
\$849 RETAIL!



Earth Computer Technologies, Inc.  
P.O. Box 7089  
Laguna Niguel, CA 92607  
Phone: (714) 448-9368  
Fax: (714) 448-9018

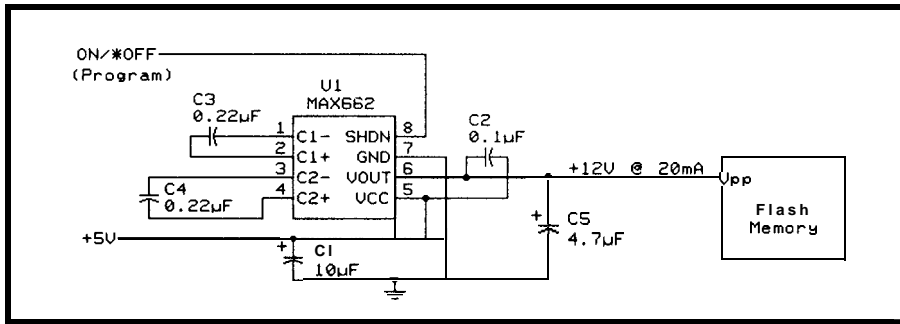


Figure 7—Twelve volts of power is supplied by dedicated DC-DC converters to program flash memory in such instruments as palmtop computers and stand-alone data loggers.

designers because they are small, readily available, and easy to connect to. A MAX639 provides a very elegant way of getting +5 V out of one of these. Using a 9-V alkaline battery with the circuit of Figure 4, more than 40% extra battery life is possible using the

5 A with efficiencies in the 80–90% range.

Not all circuits in an application require +5 V, and DC-DC converters are available to produce other voltages. For example, some flash-memory programming requires +12 V, LCD-

flash memories. As shown in Figure 8, 0 to -24-V LCD-bias voltage can be obtained using a MAX759 from a +5-V input. Symmetrical output is easily obtained using the industry-standard ICL7660 operating within the circuit in Figure 9.

Finally, full-function portable power ICs, such as Maxim's MAX714, '715, and '716 chips, combine multiple-voltage DC-DC regulators, backup-battery switchover, and microprocessor-supervisory functions in a single package to simplify the design and manufacturing of battery-powered, microcomputer-based instruments.

These are only a few examples of the many possibilities available to the designer. Check data books and

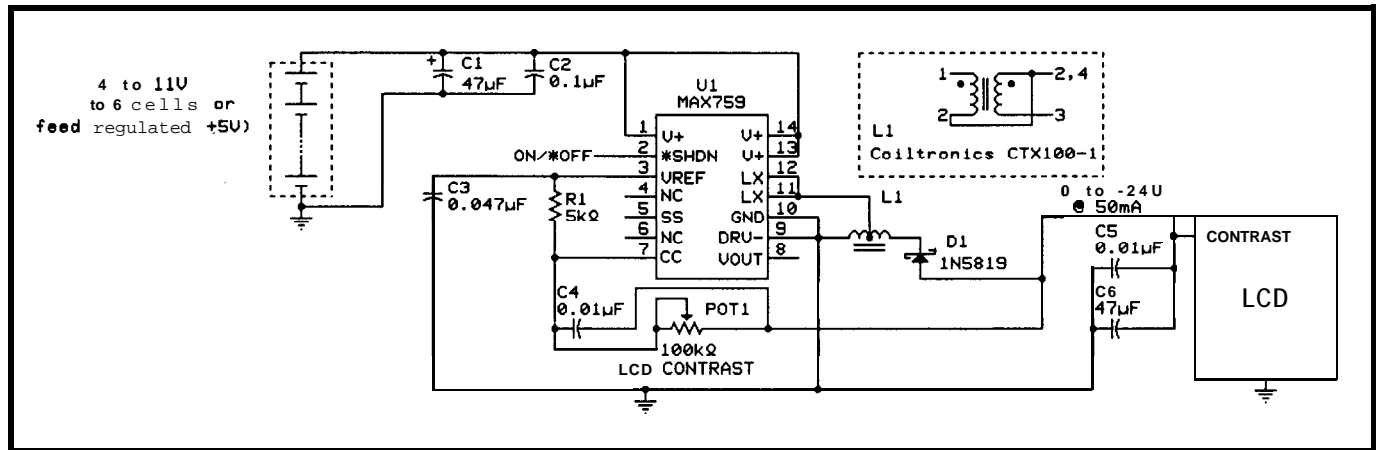


Figure 8—Negative LCD contrast voltage can be generated using this DC-DC converter with 0 to -24-V output

MAX639 instead of a low-dropout linear regulator!

For higher power requirements in instruments which integrate a complete microcomputer, high-efficiency step-down switching regulators can be used to convert the power of 6-8 NiCd or NiMH cells in series to +5 V. The circuit in Figure 5 reaches 93 % efficiency, providing a maximum current of up to 2.5 A. If you have higher current requirements, the 10–40 V obtained from sealed-lead-acid batteries can be used with a MAX724 high-power step-down DC-DC converter as shown in Figure 6. This power IC can supply up to

display contrast controls require voltages down to -24 V, and bipolar A/D and D/A converters require symmetrical power.

The circuit in Figure 7 makes use of the MAX661, a 5-12-V input DC-DC converter, which produces a regulated +12-V output to program

application sheets provided by micro-power DC-DC converter manufacturers for more information.

## BACK-UP BATTERIES

Back-up batteries are normally used to maintain volatile memory and real-time clock devices which operate

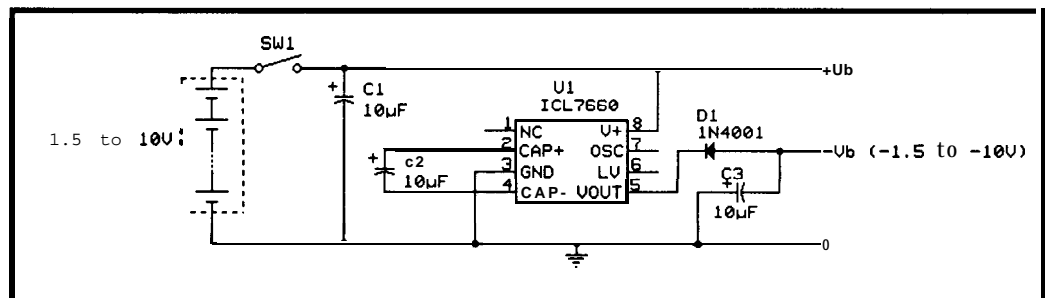


Figure 9—This simple circuit creates a symmetrical voltage about ground to power dual-supply analog circuits, as well as bipolar A/D and D/A converters.

after the main power has been, willingly or accidentally, turned off. Figure 10 shows three possible methods the designer can choose to keep a device powered.

The first makes use of a primary battery, which must be isolated using a diode to prevent charging the battery by the main supply. The second option involves using a rechargeable battery, which is trickle charged by the main power supply. The last option uses a very large (capacitance-wise) capacitor as the back-up energy storage device.

With the recent advent of the small 5.5VDC capacitors with super high capacitance (up to 3.5 F), the latter option becomes the most convenient. They are highly reliable, compact, and have superior stability over repeated charge and discharge cycles. Rechargeable batteries work in this application very much like a large capacitor, but a simple trickle-charge circuit can hardly take advantage of the battery's full life, which leads to relatively low reliability.

The use of primary batteries is the most popular today, not only for the simplicity of the design, but also because they permit a much larger period of data retention. When used with a modern micropower CMOS RAM, an alkaline or lithium battery maintains data for a period which approaches its shelf life, while a capacitor or a NiCd battery requires periodic charging.

## Battery Glossary

**Bulk Charging Rate**-current flow through a secondary battery forced by the charger during the fast-charging part of the cycle.

**Capacity (C)**-the ability of a fully charged battery to supply current over a certain time period. Capacity is measured in ampere-hours [Ah]. Capacity, as marked on batteries, is usually defined for a battery-discharge rate of  $C/10$  at a constant 20°C temperature. Capacity changes drastically as a function of discharge rate, temperature, and state of charge.

**Discharge Voltage Regulation**-stability of the voltage across the battery as a function of discharge. For example, zinc-carbon, alkaline, and lead-acid batteries slowly decrease in voltage as they discharge. There are others, however, like NiCd, mercury, and lithium that maintain an almost constant operating voltage over the entire discharge cycle, with a sudden drop at the end.

**Float Voltage**-voltage at which a rechargeable battery cell may be held indefinitely without damage.

**Overcharge Voltage**-voltage which must be reached during the charge cycle of a secondary battery to achieve full capacity.

**Primary and Secondary Batteries**-Primary batteries are used only once while a secondary battery is a storage device that can be discharged and recharged to its full capacity many times.

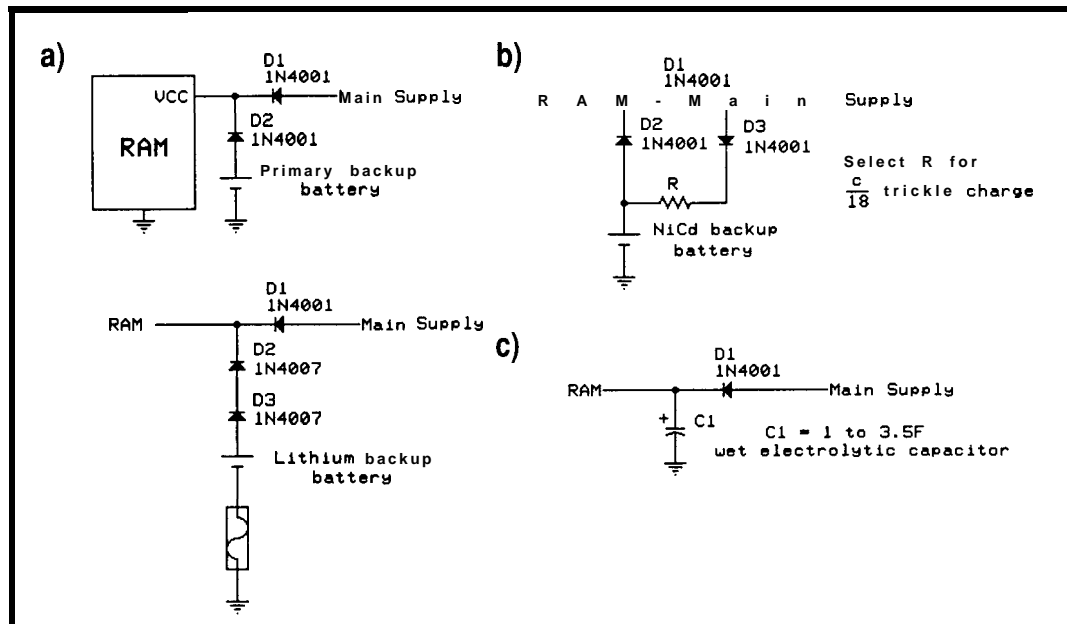
**Self-Discharge**-is the rate at which an idling (or stored) battery loses capacity as a function of temperature. For example, an unattended NiCd battery loses all of its charge in about 3 months, while a solid-state lithium battery has a shelf life of more than 20 years @ 70°C.

Most primary batteries can be used for backup. However, if the application demands the use of lithium technology, it is better to isolate the battery from charging currents using two diodes in series as well as a low-current fuse on the ground side. If noise-free back-up battery switching is required, a glitchless, high-performance switch-

over can be implemented using a dedicated IC such as the MAX1259 battery manager.

Another type of battery backup is required for applications in which surge currents can exceed battery capacity. Such surges can weaken the battery so much that power supply circuits stop operating, causing complete system failure. For example,

Figure 10—There are several backup-power options for volatile memories and CMOS microprocessors including a) primary batteries, b) secondary batteries, and c) large energy-storage capacitor.





rundown primary batteries are often unable to supply the peak power demands of EEPROM programming or electromechanic actuation, although their capacity suffices for the otherwise normal operation of the system.

As well, miniature super-high-capacitance electrolytics do not help in this situation because their internal resistance is high enough to make them incapable of delivering high currents. In these cases, one solution is to use a small NiCd battery as a capacitor to handle surges.

## FUSES, CIRCUIT BREAKERS, AND WIRING

Fuses and circuit breakers used in battery-powered instruments must be sized to protect not only the electronic circuits, but also the main power wires and the battery itself.

Primary-battery-powered supplies generally provide low currents, and for this reason, fusing is seldom used in their circuits. However, most secondary batteries have extremely low internal impedance. For example, a NiCd D-size cell delivers up to 50 A for short periods. Currents of this magnitude make these cells severe fire hazards under short-circuit conditions.

As a rule of thumb, many designers choose a fast-blow fuse or circuit breaker rated twice the maximum expected current draw. A slow-blow fuse (or breaker with long trip delay) should be used to protect circuits that require high instantaneous currents.

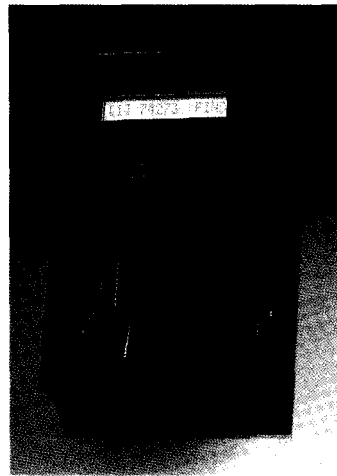
Wire sizes should be chosen to withstand currents larger than those handled by the fuses. The wire's current-handling characteristics depend on its gauge and the thermal rating of its insulation. Table 4 provides a convenient method for wire size selection.

## BATTERY CASING

Modern sealed batteries designed for electronic applications are far more reliable and better sealed than early transistor-radio batteries. They seldom ooze corrosive fluids. Nevertheless, damage to equipment in the near and not-so-near proximity of the battery is still a real possibility which must be accounted for.



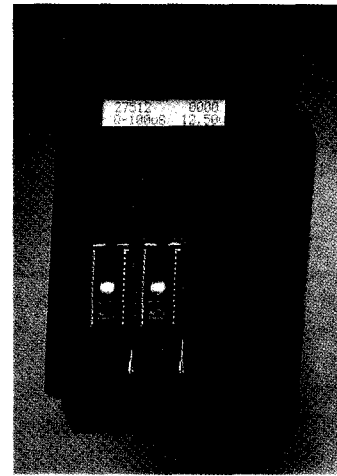
Data Genie offers a full line of test & measurement equipment that's innovative, reliable and very affordable. The "Express Series" of stand-alone, non-PC based testers are the ultimate in portability when running from either battery or AC power. Data Genie products will be setting the standards for quality on the bench or in the field for years to come.



**HT-28 Express**

### DIGITAL IC TESTER

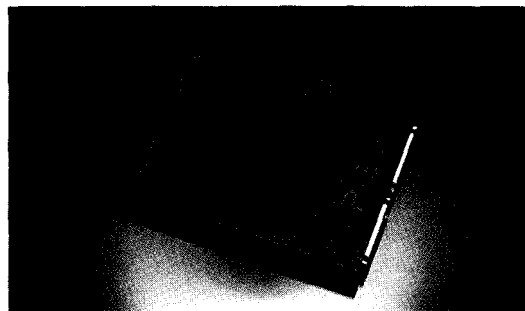
The HT-28 is a very convenient way of testing Logic IC's and DRAM's. Tests most TTL 74, CMOS 40/45 and DRAM's 4164-414000, 44164-441000. It can also identify unknown IC numbers on TTL 74 and CMOS 40/45 series with the "Auto-Search" feature.  
\$189.95



**HT-14 Express**

### EPROM WRITER

The HT-14 is one-to-one EPROM writer with a super fast programming speed that supports devices from 27328 to 27080. with eight selectable programming algorithms and six programming power (VPP) selections.  
\$289.95



**P-300**

### PC ADD-ON PROTECTOR

The Data Genie P-300 is a useful device that allows you to quickly install add-on cards or to test prototype circuits for your PC externally. Without having to turn off your computer to install an add-on cards, the P-300 maintains complete protection for your motherboard via the built-in current limit fuses.  
\$349.95

**MING**  
Microsystems  
Division of MING & P. INC.™  
17921 Rowland Street  
City of Industry, CA 91748  
TEL : (818) 912-7756  
FAX : (818) 912-9598

Call for a dealer near you.  
**1-800-473-6606**

Data Genie products are backed by a full year limited factory warranty.

Wire Gauge	150°C		175°C	
	Max. Current Rating	Min. for Fuse Rated [A]	Max. Current Rating	Min for Fuse Rated [A]
26	12	4	14	4
24	13	5	15	5.5
22	18	7	20	7.5
20	24	9	27	10
18	35	14	39	15
16	38	17	42	19
14	56	23	62	25
12	62	33	68	36
10	80	42	90	48
8	120	60	130	69
6	170	87	180	95
4	220	120	250	145
2	270	160	300	180
0	370	240	405	280

Table 4—Wire-size selection is very important in fuse-protected, battery-operated equipment. Minimums are the absolute lowest recommended values and may cause considerable heating of wires.

Electronic circuits of small, primary-battery-powered instruments can usually be protected by keeping the batteries in a plastic battery compartment. For this reason and the convenience, custom-made enclosures for consumer products often incorporate an isolated battery compartment.

The same approach can be used by the experimenter. A layer of RTV silicone sealant can be applied to the walls of the battery compartment to better protect electronic circuits from eventual battery leakage. Also, it is a good idea to use a good-quality battery holder or battery clip because the white deposits that alkaline and mercury batteries develop on their terminals damage the connectors, making them unreliable.

Regardless of the rather liberal design considerations for most primary batteries, the casing of lithium batteries deserves great consideration because of the explosive hazard they pose. The enclosure of secondary batteries should be of greater concern to the experimenter because the risks involved are chemical and physical. The current these batteries deliver can cause wires to heat to the point that a fire hazard exists.

Although modern sealed-lead-acid, NiCd, and silver-zinc batteries incorporate leak-proof seals and cases, they seldom can handle the enormous pressure and heat that develops under flawed charging (e.g., a high-rate overcharge) or discharging (e.g., a continuous short with full charge).

Good design calls for the use of an isolated battery box in which all of the cells of the battery are firmly secured. All interior surfaces of the battery box must be coated with a nonconductive, electrolyte-resistant material. All unused space must be filled with absorbent material in case the electrolyte leaks. The box must be strong enough to be able to contain an explosion. In addition, if the box is constructed of a conductive material, the main battery fuses should be placed inside the box.

Rechargeable batteries often produce corrosive and/or combustible gases which must be vented to avoid concentrating them to the point where an explosive mixture is generated. Instruments built for outdoor applications frequently use a water-tight enclosure. In these, adequate venting of gases to the atmosphere must be ensured. This is especially compelling when lead-acid batteries are used because hydrogen and oxygen mixtures produced during charging are highly explosive if inadequately diluted.

#### EVERY DAY LIFE

The use of batteries in portable electronic equipment is ever more important in this day of cellular phones, laptops, and embedded controllers. Knowing what kinds of batteries are available and how to design with them can spell the success or failure of a product on the market. I hope I've provided you with a good starting point for your next design. □

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. C. D. S. Tuck, CDS: **Modern Battery Technology** (New York: Ellis Horwood Ltd., 1991) 87-185.
2. P. Horowitz and W. Hill. **The Art of Electronics**, 2nd ed. (Cambridge: Cambridge University P, 1989) 924-926.
3. Gates Energy Products, Inc., **Rechargeable Batteries Applications Handbook** (Boston: Butterworth-Heinemann, 1991).
4. Maxim Integrated Products, **MAX712 and MAX713 NiCd/NiMH Fast-Charge Controllers**, Data Sheet 19-0100 (October 1992).
5. D.W. Potter, "Those NiCad Batteries and How to Charge Them!" **QST** October 1981: 34-35.
6. K. Stuart, "Getting the Most Out of Nickel-Cadmium Batteries," **QST** February 1992: 40-45.
7. Unitrode Corporation, "UC2906 and UC3906 Data Sheet," **Linear Integrated Circuits Data and Applications Handbook**, IC600, (1990), 4.219-4.229.
8. W. Dion, "A New Chip for Charging Gelled-Electrolyte Batteries," **QST** June 1987: 27-29.
9. Unitrode Corporation, "Applications Note: Improved Charging Methods for Lead-Acid Batteries Using the UC3906," **Linear Integrated Circuits Data and Applications Handbook**, IC600, (1990) 9.87-9.97.

413 Very Useful  
414 Moderately Useful  
415 Not Useful

# DEPARTMENTS

50 Firmware Furnace

60 From the Bench

68 Silicon Update

76 Embedded Techniques

82 ConneCTime

## FIRMWARE FURNACE

Ed Nisley

# Journey to the Protected Land: Infrastructure Improvement



It's time to put some punch into those tiny

taskettes Ed introduced last month. Taking advantage of protected mode, he adds a task dispatcher to handle traffic control while each task goes about its own business.



years ago, a friend recommended a surefire cure for my '74 Valiant's engine

troubles: "Jack up the radiator cap and drive a new car underneath." Nearly a decade later, I sold that Turkey Valiant for \$400; both parties to the transaction thought they got a great deal.

In our last installment, we left a pair of trivial taskettes swapping control back and forth in a corner of the vast expanse of RAM above 1 MB. This month, we'll jack up the taskettes, drive a whole new support structure underneath, and lower them gently on a firm foundation.

Each taskette will get a separate Local Descriptor Table (LDT) with unique segments, including data segments allocated on the fly. The task dispatcher will sprout several new features, including the ability to handle an arbitrarily large number of tasks and a video status display.

If we call our job *information infrastructure improvement*, we can also be 100% buzzword compliant.

### BUILDING AN ISOLATION WARD

It should come as no surprise that the '386 CPU isolates protected-mode tasks using segmentation. Restricting each task to a unique set of segments ensures it cannot "reach" storage used by another task. The same CPU

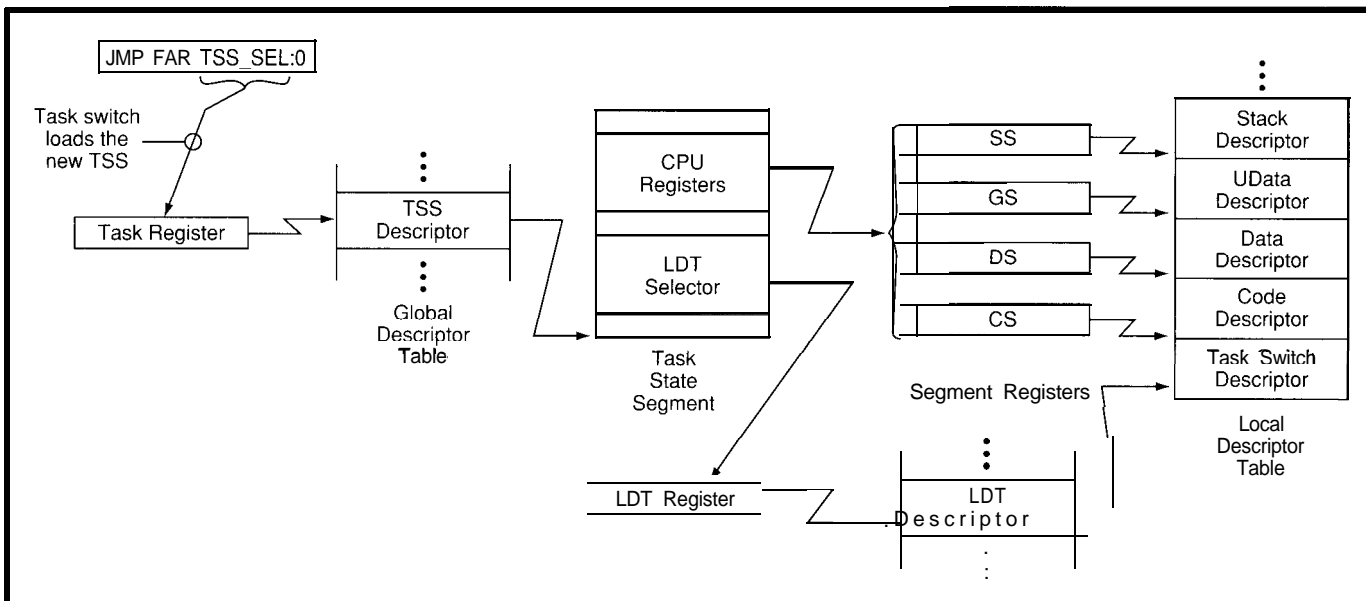


Figure 1--The CPU's Task register contains the selector for the current Task State Segment's (TSS) descriptor. When the CPU switches to a task, it loads the Local Descriptor Table (LDT) register and the CPU segment registers from the new TSS. Each task can have a separate LDT with unique code, data, and stack segments. The CPU includes privilege level checks that prevent a task from loading Global Descriptor Table (GDT) selectors, although the FFTS code has not activated that hardware yet.

hardware preventing an errant program from corrupting its own code works equally well for tasks. All we need to do is set things up properly to put that CISC CPU on our side.

The key to the isolation ward is a Task State Segment (TSS) field that we left unused last month--the task's LDT selector. The CPU loads this field into its LDT register during every task switch. By definition, all tasks have access to Global Descriptor Table (GDT) descriptors, but only the current task may directly access its LDT. If the LDTs are unique and their segments don't overlap, the task is pretty well bottled up.

The alert reader will find no reason why a task cannot misuse a GDT descriptor. For example, the GDT's data alias descriptor grants the ability to scribble all over the system's GDT, a kamikaze gesture that a viral task might enjoy.

The '386SX plugs this hole in grand CISC style by assigning a privilege level to every segment. Each descriptor includes a two-bit field defining the Descriptor Privilege Level (DPL) and a similar field in each selector sets the Requestor Privilege Level (RPL). The DPL field of the current code segment (the one in the CS register) sets the task's Current Privilege Level (CPL).

Every time an instruction loads a segment register, the CPU hardware verifies that the task is privileged enough to use the corresponding descriptor. The most privileged code runs at Level 0, the least privileged at Level 3, and there are two intermediate levels for special purposes.

You may think the levels are numbered backwards, and I'd have to agree. The following strangled sentence from the Intel documentation demonstrates the confusion caused by inverting the obvious numeric values: "Instructions may load a segment register only if the DPL of the segment is the same or a less privileged level (greater privilege number) than the less privileged of the CPL and the selector's RPL."

Got that?

OK, quickly now, if CPL = 3, DPL = 2, and RPL = 0, may the task use the segment?

Basically, here's how it works. "Most privileged" kernel code has unlimited access to all system facilities, including all GDT descriptors. Lowly tasks have a "least privileged" level matching their LDT descriptors. When a task attempts to access a "more privileged" GDT descriptor, the hardware triggers a protection exception, and the operating system terminates the offending task.

GDT descriptors may have any privilege level. For example, your code might require a shared data segment available to any task. A "least privileged" GDT descriptor covering that chunk of storage grants unrestricted access with no interference from the hardware privilege checks.

I will punt the entire subject of privilege levels until we begin working with tasks in Virtual 86 mode. Splitting the Firmware Furnace Task Switcher (FFTS) code into various privileges introduces several side effects that are devilishly hard to explain without more background, and this is hard enough already.

FFTS will, however, be structured along the general lines required by the privilege hardware. For example, the tasks use call gates to access system code even though both the caller and callee are both "most privileged" code. It isn't exactly safe computing, but at least we know where our code has been....

Some manuals describe the protection-privilege levels as *rings*. Typically, there's a figure showing a bull's-eye target with kernel code at the center and user code on the periphery. Gaining access to "Ring 0" in, say, Windows is a coveted achievement because you can do all manner of interesting and dangerous things.

When you understand how GDTs, LDTs, and privilege levels work you will understand why access to Ring 0 is hard to come by.

But, before all that, we must create an assortment of unique segments for our tasks.

## SEGMENT SELECTION

Every nontrivial task requires at least a code segment, stack segment, and data segment. If the task uses initialized variables, as most tasks do, the FFTS task set-up code must create the data segment and copy the initial values from the binary diskette image to the proper addresses. The task may use other segments requiring additional set up, as well.

Figure 1 shows how the various selectors, descriptors, and tables fit together for a single task. The CPU's Task register and LDT register hold the current task's TSS and LDT selectors. The segment-register fields in the TSS contain selectors referring to LDT descriptors. Those descriptors cover the storage segments dedicated to the task.

In *INK 52*, I described how the assembler, linker, and Locate work together to create the kernel segments using the small memory model. The protected-mode start-up code generates GDT segments that match the usual real-mode assumptions and allow us to use ordinary simplified segment directives. Refer back to that column to refresh your memory of segment names, combination attributes, classes, groups, and suchlike if the following discussion is too terse for comfort.

Our tasks, sad to say, cannot use the small model because each task must have its own code segment. The kernel's single code segment holds all of the instructions from all of the source files with offsets relative to the start of the combined segment. The code offsets within each task must be relative to the start of their unique segment, not the kernel segment, thus they cannot be combined with the kernel's segment.

Apart from that restriction, though, the small model is ideal for tasks. The larger memory models

**Listing 1--** Each task must have unique code, data, and stack segments. The standard small model creates FAR\_DATA and FAR\_BSS segments for each source file. The DefTaskCS macro mimics the separate code segment used in larger memory models without affecting the default near branches in the small model. The three LABEL directives at the bottom of the listing mark the length of each segment.

```

TestUFarData    UFARDATA
                DD      ?
                FARDATA

DemoMsg         DB      'Demo1 FAR data',NL
                DB      EOS
;--- endless task loop

                DefTaskCS

                PROC    TaskProc

                MOV     EDX, SYNC_ADDR

@@Again:
                IN      AL, DX
                OR      AL, 02h
                OUT     DX, AL
                AND     AL, NOT 02h
                OUT     DX, AL

                SysCall LDT_SWITCH

                JMP     @@Again
                ENDP    TaskProc
                EndTaskCS

;--- mark the end of each segment

                UseTaskCS
                LABEL   CodeLength
                EndTaskCS

                FARDATA
                LABEL   FarDataLength BYTE

                UFARDATA
                LABEL   FarBssLength BYTE

```

automatically create a separate code segment for each source file, which is good. Within those files, however, the assembler defaults to **FAR** JMPs and **CAL** Ls, which use the real-mode segment addresses rather than our PM selectors.

So, I used the small model with a few macros to create a unique code segment for each task file. The segment names are the file name suffixed with `_text` in the same manner as the larger memory models. For example, the code in `demo1.asm` resides in the segment named `demo1_text`.

The intricacies of MASM and TASM macro programming are well beyond the scope of this column. Refer to Jim Mischel's *Macro Magic with Turbo Assembler* for guidance in this

arcane field. Simple macros are tricky enough, complex macros keep you awake at night, and just thinking about Jim's macros gives me the shakes.

Small model programs normally access the default near data segment using the DS register. They also have access to a pair of far data segments named **FAR-DATA** and **FAR\_BSS**, selected with the **FARDATA** and **UFARDATA** directives, respectively. The former holds up to 64 KB of initialized variables and the latter holds up to 64 KB of uninitialized variables. These two segments sport the **PRIVATE** combination attribute, preventing the linker from combining them with segments from other source files. That's exactly what we want for the task's data segments!

**Listing 2**—Each task file contains a *STRUC* similar to this one summarizing its storage requirements. The *Code5 t a r t* value supplies the initial value of *EIP* in the task's TSS. The initialization code also copies *Task Name* into the TSS as an easy-to-read identifier. The real-mode segment and offset addresses provide enough information to create the PM addresses used in the PM segment descriptors.

```
SEGMENT _protconst

LABEL cInfo_Demo1 TASKINIT
TASKINIT {CodeSegment = SEG TaskProc,
          CodeStart = OFFSET TaskProc,
          CodeLength = OFFSET CodeLength,
          FarDataSegment = SEG FarDataLength,
          FarDataLength = OFFSET FarDataLength,
          FarBssLength = OFFSET FarBssLength,
          TaskName = "Demo Task 1"

ENDS _protconst
```

There is one potential complication—data accesses must use different segments depending on whether the variable is initialized or uninitialized. Keeping track by hand of which variable is where can be a major headache. The assembler is bright enough to handle the bookkeeping if we tell it which register points at each segment.

The normal small model setup uses DS for the near data in *DATASEG*, and the two far data segment selectors must be loaded as they're needed. In our application, the tasks will not have access to the near data in the kernel's segments. Given the limited number of segment registers, there is no reason to waste one on inaccessible data.

The *DefTaskCS* and *UseTaskCS* macros in Listing 1 include *ASSUME* directives that associate a segment with each of the segment registers. The CS register, of course, selects the source file's unique code segment. DS, normally pointing at the near data in *DATASEG*, is redirected to *FAR-DATA*. GS, one of the additional segment registers available on '386 and higher CPUs, selects the task's *FAR\_BSS*. The assembler automatically uses the register corresponding to the variable's segment and generates a segment override opcode if necessary.

Although it's not strictly proper, each task also has access to the kernel's constant-data segment through the FS register. A truly secure system will not allow even read-only

access to kernel information. In our case, the tasks are basically friendly and won't attempt to peek at data they don't own. They can't overwrite it in any case.

Should you feel the need for a separate constant segment in each task, use macros similar to those in Listing 1 with a smidge of task set-up code to create a read-only data segment. As with the kernel segment's constants, you need not copy them to a different location because they're never changed.

Apart from the macros around the code segment, Listing 1 resembles any ordinary real-mode program. Study the linker map and *Locate*'s output when you reassemble these file to see how they arrange all the segments.

## SLICING STORAGE

Before a task is ready for its first switch, the FFTS kernel must create GDT and LDT descriptors for each of its segments, set up the TSS, and prepare the initialized variables. The *TASKINIT STRUC*, shown in Listing 2, provides all the information FFTS needs to get the addresses and sizes right.

Because each task is contained entirely within a single source file, the labels at the bottom of Listing 1 mark the end of their respective segments and, thus, the segment lengths. The linker inserts padding between the data at the end of one segment and the start of the next segment to ensure

each segment begins on a paragraph (16-byte) address boundary. The length of the protected-mode segment excludes that padding to prevent use of any "unauthorized" storage.

The best way to see what FFTS does while creating a task is to study the big loop shown in Listing 3. Prior to this loop, the kernel has just initialized its own TSS to receive the state of the CPU during the first task switch. The loop starts out by creating *pTaskInit*, *apointertocTaskTable*, which is an array of pointers to the *TASKINIT* structures that are shown in Listing 2.

As you saw last month, TSSs form an array in the *GDT\_TSS\_ALIAS* segment. The *pTSS* variable points to the current TSS in that array; *pTSS - Linear* holds the equivalent linear address required in descriptor base address fields.

Each TSS in the array requires two GDT descriptors: one for the TSS and a second for the LDT. For simplicity, each task's descriptors are consecutive in the CDT, making the LDT selector numerically equal to the TSS selector plus 8. For example, I defined the first task's TSS selector as 1000 (hex) and its LDT selector as 1008. This makes the task selectors easy to remember: 1000, 1010, 1020, and so on.

The *MemSetDescriptor* function initializes a descriptor in the system GDT or a task's LDT. It uses the descriptor's selector number as an index into a data-alias segment covering the descriptor table to write the fields. Each LDT requires a separate data alias, so I wrote *MemMakeDataAlias* to copy a descriptor's contents into an unused GDT entry, set its flags to allow read-write data access, and return the alias's new selector.

The descriptor in *LDT[0]* is a new item: a *cull gate*. The CPU protection hardware prevents a "least privileged" task from calling a "most privileged" kernel routine directly. Call gates are the approved way to pass control between privilege levels while preserving protection. They will become vital when our tasks run at Level 3 instead of Level 0. We might as well prepare for that day.



As you can see from MemSet Descriptor's parameters, a call gate contains the segment selector and offset address of the privileged kernel routine. In this case, the gate grants access to the kernel's TaskDispatch routine. TaskDispatch uses no parameters, a simplification that I'm grateful for because I'm running out of space in this column. We'll cover the general case in a month or so.

The SysCall macro in Listing 1 creates a FAR CALL instruction with the LDT\_SW I T C H selector in the segment part of the address. When the CPU executes this instruction, it reads the L D T \_ S W I T C H selector, extracts TaskDispatch's code-segment selector and offset from the descriptor, and begins execution at that address. When TaskDispatch executes a RET instruction, the CPU unstacks the return address and resumes execution immediately after the SysCall.

Call gates may live in the GDT where they are accessible to all tasks that meet their privilege requirements or in an LDT for use by a single task. I put the TaskDispatch gate in the LDT primarily to demonstrate that you can do it this way. One application for an LDT gate is providing different task dispatchers for various types of tasks. Each LDT would contain a gate to the appropriate dispatcher even though the tasks would use the same SysCall branch with the same LDT\_SW I T C H selector.

A call gate restricts access to a single privileged entry point in the kernel's code segment. The task cannot access either the GDT or LDT descriptors as data (at least when the proper privilege levels are in effect!) and, thus, cannot alter the gate descriptor's contents. The only two pieces of information the task needs to use a kernel routine are its call-gate selector and the proper parameters.

The remaining LDT descriptors are more familiar. Creating the code-segment descriptor in LDT[1] requires converting the task's real-mode segment address into the corresponding protected-mode linear address. The linker and Locate adjust the segment addresses based on the final location of the code in the binary image. Multi-

plying the segment by 16 (by shifting left four bits) and appending 12 low-order zeros gives the segment's starting byte address. The FFTS diskette loader puts the binary image at 1 MB, which means the final linear address is 1 MB plus the real-mode byte address.

The data segment in LDT[2] requires a similar computation. In this case, however, the FFTS start-up code has already copied the block of far data from the binary image to a separate segment covered by GDT\_TASKDATA. The linear address of the task's data is the PM block's starting address plus the difference between the real-mode addresses of the task's segment and the far data class.

The uninitialized data segment in LDT[3] and the stack segment in LDT[4] are easier. All the task needs is a segment-register value pointing to an appropriately sized chunk of storage. I

wrote MemAllocPerm to carve the vast expanse of storage above the FFTS kernel into useful chunks.

The FFTS kernel extends from the 1-MB line up through the end of the 64-KB block reserved for the task initialized-data segments. All of the remaining storage between 00150000 and the end of RAM falls into a single segment called GDT\_MALLOCC. There will eventually be two memory allocators at work in that block: MemAllocPerm, which assigns "permanent" segments from the top down, and MemAlloc, which parcels out temporary segments from the bottom up.

A header preceding each new block of memory describes the contents. I include the creator's task ID, a short string comment, and a serial number along with the usual block address, length, and so forth. There's no way to display this infor-

*Listing 3—This code steps through an array of TASKINIT structures and extracts the values needed for the TSS, LDT, segments, and register contents of each task. The FFTS startup code has already copied the initialized variables in all of the tasks' FAR\_DATA segments to the GDT\_TASKDATA segment. This code creates a data segment covering only a single task's contribution to that block. The uninitialized data and stack segments are allocated from RAM above the kernel's fixed locations.*

```

... shorthand for the standard pointers
TSS_PTR EQU <<(TSS_PTR ES:EDI)>
INIT_PTR EQU <<(TASKINIT_PTR FS:ESI)>

<<< considerable code omitted >>>

MOV [pTaskInit],OFFSET cTaskTable
SUE [pTaskInit],4 ; correct for pre-inc

@@InitLoop:
ADD [TSSsel],TSS_DESCSTEP; step to next selector
ADD [pTSS],TSS_SPACING ; ... and TSS entry
ADD [pTSSlinear],TSS_SPACING
ADD [pTaskInit],4 ; ... and table entry

MOV ESI,[pTaskInit] ; get pointer to table entry
MOV ESI,[FS:ESI] ; get pointer to init info
CMP ESI,0 ; if zero, we're done
JE @@InitDone

; show task name and copy the string into the TSS
LEA EAX,[INIT_PTR.TaskName]

<<< display code omitted >>>

MOV EDI,[pTSS] ; set up ES:EDI as TSS ptr

LEA EBX,[TSS_PTR.TaskName]
CALL StrNCopy,GDT_TSS_ALIAS,EBX,TASKNAME_SIZE,\
GDT_CONST, EAX

```

(continued)

```

; create the TSS descriptor and pop it into the dispatching array

CALL MemSetDescriptor,[TSSSel],GDT_GDT_ALIAS, \
    [pTSSLinear], SIZE TSS, ACC_TASK32, 0

MOV EAX,[TSSSel]
MOV EBX,[DispIndex]
MOV [(DispArray.TaskPtr.Seg) + EBX*SIZE DISP_ENTRY],AX
MOV [(DispArray.TaskInfo) + EBX*SIZE DISP_ENTRY], \
    (MASK TaskPresent) + MASK TaskDispatchable
INC [DispIndex]

; create LDT selector, descriptor, and LDT data alias

MOV EAX,[TSSSel]
ADD EAX,SIZE DESC_NORM
MOV [TSS_PTR.LDTSel],AX

MOV EBX,[pTSSLinear]
ADD EBX,OFFSET (TSS PTR 0).TaskLDT
CALL MemSetDescriptor,EAX,GDT_GDT_ALIAS, \
    EBX,TASKLDT_SIZE*SIZE DESC_NORM,ACC_LDT,ATTR_32BIT

MOVZX EAX,[TSS_PTR.LDTSel]
CALL MemMakeDataAlias,EAX,GDT_GDT_ALIAS
MOV [LDTAlias],EAX

; set up task switch gate in LDT[0]

CALL MemSetCallGate,LDT_SWITCH,[LDTAlias], \
    GDT_CODE,OFFSET TaskDispatch,ACC_CALLGATE,0

; set up CS:EIP and create code segment in LDT[1]

MOV [TSS_PTR.GS],LDT_CODE
MOV EAX,[INIT_PTR.CodeStart]
MOV [TSS_PTR.EIP],EAX

CALL MemGetDescBase,GDT_CODE,GDT_GDT_ALIAS ; our CS
MOVZX EBX,[INIT_PTR.CodeSegment] ; convert task seg
SHL EBX,4 ; to linear
ADD EAX,EBX ; code seg + task seg
CALL MemSetDescriptor,LDT_CODE,[LDTAlias], \
    EAX,[INIT_PTR.CodeLength],ACC_CODE32,ATTR_32BIT

; set DS and create the task's data segment in LDT[2]
; the startup code copies the initialized values into GDT_TASKDATA
; so all we do is set the starting point and length of our data

MOV [TSS_PTR.DS],LDT_DATA

MOV BX,[INIT_PTR.FarDataSegment]; far data seg base
SUB BX,SEG PMFarDataBase ; far data class base
MOVZX EBX,BX
SHL EBX,4 ; convert to linear
CALL MemGetDescBase,GDT_TASKDATA,GDT_GDT_ALIAS
ADD EAX,EBX

MOV EBX,[INIT_PTR.FarDataLength]

CALL MemSetDescriptor,LDT_DATA,[LDTAlias], \
    EAX,EBX,ACC_DATA32,ATTR_32BIT

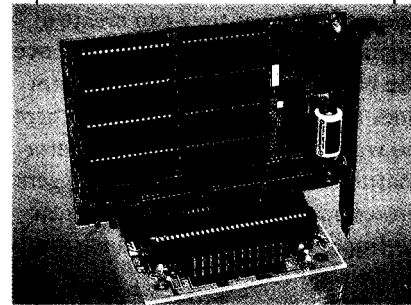
; set GS and allocate the task's uninitialized data seg in LDT[3]
; if allocation fails the task fails with a data segment error

MOV [TSS_PTR.GS],LDT_UDATA

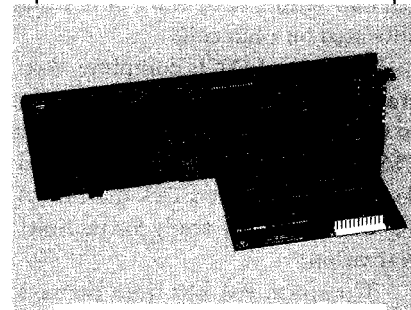
```

(continued)

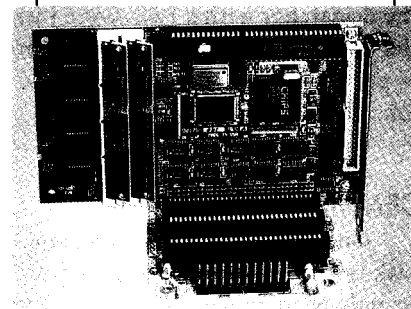
**VMAX**<sup>®</sup>  
**QUALITY PRODUCTS**  
**RESPONSIVE SERVICE**  
**RELIABLE DELIVERY**

**SOLID STATE DISK — \$135\***

½ Card 2 Disk Emulator  
 EPROM, FLASH and/or SRAM  
 Program/Erase FLASH On-Board  
 1M Total, Either Drive Bootable

**25MHZ 386DX CPU — \$695\***

Compact AT/Bus or Stand Alone  
 On-Board SVGA, IDE, FDC, 2 Ser/Bi-Par  
 FLASH/SRAM Drives to 2.5M  
 Cache to 128K, DRAM to 48M

**TURBO XT  
w/FLASH DISK — \$266\***

To 2 FLASH Drives, 1M Total  
 DRAM to 2 M  
 Pgm/Erase FLASH On-Board  
 CMOS Surface Mount, 4.2" x 6.7"  
 2 Ser/1 Par, Watchdog Timer

All Tempustech VMAX products are  
 PC Bus Compatible. Made in the  
 J.S.A., 30 Day Money Back Guarantee  
 \*QTY 1, Qty breaks start at 5 pieces.

**TEMPUSTECH, INC.****TEL: (800) 634-0701****FAX: (813) 643-4981**

Fax for 295 Airport Road  
 fast response! Naples, FL 33942

mation yet, but we'll be glad it's there when we need it!

You have surely seen this method of memory allocation in real mode. The nice thing about protected-mode allocators is that they can create a segment descriptor that exactly covers the block of storage. The descriptor does *not* include the header, so there is no way to corrupt the kernel's information. As always, the CPU triggers a protection exception when the task attempts to write beyond the bounds of the segment.

I'll have more to say about memory allocation later on when the tasks need dynamic memory. An obvious application will be replacing the static message buffer in `StrFormat` with a temporary buffer that's allocated on each call.

After the LDT is complete, the start-up code sets the task's FS register to the `_p r o t c o n s t` segment, puts the TSS selector in its EAX register, and displays the finished TSS and LDT. The task is finally ready for its first activation!

Complex, yes, but from the task's point of view the setup is entirely invisible. Each task gets control on the first task switch with its segment registers preloaded, a clean stack, and all the variables it needs. Tasks invoke the dispatcher with a `FAR CALL` (using the `SysCall11` macro) and regain control as though the whole kernel is just a function.

Now, for that first task switch.. .

## CENTRAL DISPATCHING

Once you allow more than one task in your system, a decision rears its ugly head: which task to run next! There are at least as many solutions as operating systems, each optimized in a different direction.

Although the buzz phrase nowadays is "preemptive multitasking," we're going to start out with a much simpler method-cooperative multitasking with round-robin dispatching. Although it's not at the cutting edge, it will serve our purposes for quite a while.

"Cooperative multitasking" means each task runs until it returns control to the `TaskDispatch` routine.

Listing 3-continued

```

CALL MemAllocPerm,LDT_UDATA,[LDTAlias], \
[INIT_PTR.FarBssLength],ACC_DATA32,ATTR_32BIT, \
GDT_CONST,OFFSET cFarBssDesc,[TSSSel],0
; allocate task stack with LDT[4] descriptor, set SS:ESP
; if allocation fails, the task fails with a stack segment error

CALL MemAllocPerm,LDT_STACK,[LDTAlias], \
TASK_STACKSIZE,ACC_STACK32,ATTR_32BIT, \
GDT_CONST,OFFSET cStackDesc,[TSSSel],0babefaceh

MOV [TSS_PTR.ESP],TASK_STACKSIZE-4; allow unused DWOR0
MOV [TSS_PTR.SS],LDT_STACK

; set the remaining task registers

MOV [TSS_PTR.FS],GDT_CONST ; global constants

MOV EAX,[TSSSel] ; tell task its TSS
MOV [TSS_PTR.EAX],EAX

; discard the LDT alias and display the complete TSS

CALL MemKillDescriptor,[LDTAlias],GDT_GDT_ALIAS

TrcIf TrcTaskMake
CALL TaskDumpTSS,[TSSSel] ; show what we did
TrcEndIf

JMP @@InitLoop ; do another task

```

The advantage is that the task itself determines when it should yield control to the dispatcher, thus eliminating much of the need for semaphores and operating-system interlocks. The disadvantage is that a single task can fall on the ball, never call `TaskDispatch`, and wedge the entire system.

Lest this sound entirely too fragile for words, it's how Windows works. We're starting with simpler and better-behaved tasks than your average Windows application, increasing the odds that things will keep clocking along as desired.

"Round-robin dispatching" means the task dispatcher iterates through all the tasks in a fixed order. This algorithm gives every task the same priority and, not coincidentally, eliminates a lengthy discussion of priority management. Idle tasks may return to the dispatcher immediately, hardworking tasks may gnaw through a large chunk of computation, and everybody gets a share of the CPU at least once in a while.

Listing 4 shows the new `TaskDispatch` routine. The task set-up code loads `DispatchArray` with the TSS selectors and control flags as it creates each task. The kernel task occupies `DispatchArray[0]` and passes control to the task in `DispatchArray[1]` on the first task switch. The dispatcher continues through each entry until it finds a zero `TaskPresent` flag, at which point it resets `DispatchIndex` and starts over again with the kernel task.

The `TaskDispatchable` flag provides a way to skip a task that doesn't need dispatching. Some '386SX functions, such as error and interrupt handlers, require separate tasks that should not be dispatched by the operating system. This flag comes in handy next month when we explore those topics. For now, it's just a placeholder. All our taskettes are eminently dispatchable!

If your system has a video display, you can watch the task dispatcher in action. Each pass through `TaskDispatch` updates a status line on the bottom of the CRT with the current

Display, TSS selector, and cumulative number of task switches. Bit 15 of the FDB DIP switches turns this display off. Watch the FDB LED digits to see how that affects task-dispatching performance.

The LPT1 trace outputs reveal that one iteration of each trivial taskette this month requires about 40 us on a 33-MHz '386SX. Computing and writing the status values to the screen adds about 700 us to each task switch. A single pass through the four DisplayArray entries thus requires 160 us with tracing disabled and 3 ms when you watch the numbers blurring on the screen.

The video status requires an unseemly proportion of time simply because the tasks aren't doing anything. Keep this in mind when you start writing your own code...and keep that scope handy!

#### IS IT WORTH IT?

By this time, your head is undoubtedly reeling from all the TLAs (Three-letter Acronyms), segments,

selectors, descriptors, pointers, and whatnot. Indeed, the single biggest challenge of this entire series has been explaining how the myriad segments fit together. You've certainly wondered why I didn't opt for flat 32-bit PM programming—set the segments to span all of the PC's storage and be done with this nonsense!

Most of the articles and columns you've read elsewhere deal with PM programming for a particular operating system. Whether the author uses OS/2, the long-awaited 32-bit Windows, a DOS extender, or one of the \*NIX crowd, the conditions under which the code runs are fixed. If the OS sets up the segment registers so the user code runs in flatland, well, then they write flat 32-bit protected-mode code.

In our case, the FFTS is the operating system, rudimentary though it may be. If I were writing a single protected-mode program on bare silicon, we'd be done now—flip into 32-bit protected mode, set a few GDT segments, ignore memory protection, bypass multitasking, and go!

But, that treats the '386 as a glorified 8051—fast and limited. If your problem is complex enough to justify a '386 embedded controller running in protected mode, it's probably also complex enough for multitasking. If it's *that* complex, those tasks should be protected from each other. The '386 has that protection built right into the hardware.

So, here we are with all those segments. It's a dirty job, but somebody's gotta do it.

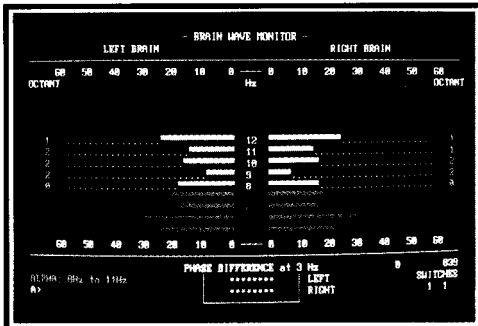
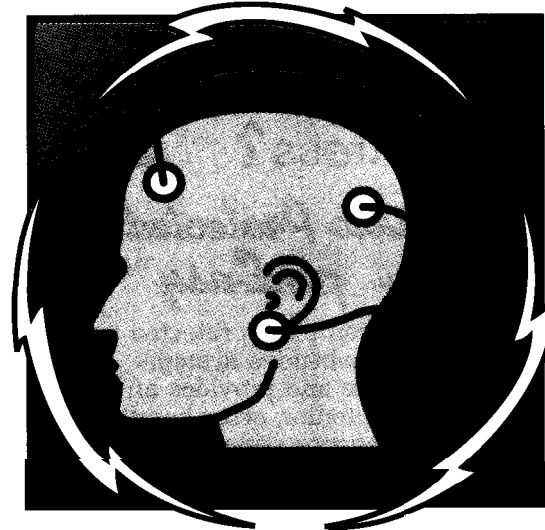
And that's why I'm going through this exercise. You can tear off whatever chunk of FFTS fits your requirements. The code in *INK 52* can serve as the basis for a flat 32-bit program if you twiddle the GDT descriptors just a bit. The code last month is the simplest possible hardware-assisted multitasker. The LDTs this month lay the groundwork for a more robust multitasker that makes the tasks easier to debug.

Just wait until we get closer to Virtual 86 mode because the CPU requires paging to run more than one

# HAL-4

## EEG Biofeedback Brainwave Analyzer

The HAL-4 kit is a complete battery-operated 4-channel electroencephalograph (EEG) which measures a mere 6" x 7". HAL is sensitive enough to even distinguish different conscious states between concentrated mental activity and pleasant daydreaming. HAL gathers all relevant alpha, beta, and theta brainwave signals within the range of 4-20 Hz and presents it in a serial digitized format that can be easily recorded or analyzed. HAL's operation is straightforward. It samples four channels of analog brainwave data 64 times per second and transmits this digitized data serially, to a PC at 4800 bps. There, using a Fast Fourier Transform to determine frequency, amplitude, and phase components, the results are graphically displayed in real time for each side of the brain.



**HAL-4 KIT..... NEW PACKAGE PRICE - \$279 +SHIPPING**  
Contains HAL-4 PCB and all circuit components, source code on PC diskette, serial connection cable, and four extra sets of disposable electrodes.

to order the HAL-4 Kit or to receive a catalog,  
**CALL: (203) 8752751 OR FAX: (203) 8752204**

**CIRCUIT CELLAR KITS • 4 PARK STREET  
SUITE 12 • VERNON • CT 06066**

-The Circuit Cellar Hemispheric Activation Level detector is presented as an engineering example of the design techniques used in acquiring brainwave signals. This Hemispheric Activation Level detector is not a medically approved device, no medical claims are made for this device, and it should not be used for medical diagnostic purposes. Furthermore, safe use requires HAL be battery operated only!

Listing 4—The task initialization code in Listing 3 creates an array of TSS selectors and control flags with the kernel task as the first entry. The kernel task calls this PROC to switch to the next task in the array and the process continues each time control returns here. Each pass reads the FDB DIP switches to control the video status output through the TrcIf and TrcEndIf macros.

```

CODESEG
PROC TaskDispatch FAR
USES EAX,EBX,EDX,DS,ECX

;--- set up DS to kernel's data to reach the dispatching variables

MOV EAX,GDT_DATA
MOV DS,AX

CALL DbgGetSwitches ; sample the switches for tracing

;--- select and dispatch the next task
; a simple round-robin run through all the tasks in the array
; we dispatch the kernel task (index = 0) every time

MOV EBX,[DispIndex] ; get current task index

@@DispSkip:
INC EBX ; step to next task in array

MOVZX EAX,[(DispArray.TaskInfo) + EBX*SIZE DISP_ENTRY]
TEST EAX,MASK TaskPresent ; entry in table?
JNZ @@DispGo ; 1 = yes, so do it
XOR EBX,EBX ; 0 = no, hit the end
JMP @@DispForce ; always run kernel!

@@DispGo:
TEST EAX,MASK TaskDispatchable ; OK to run this task?
JZ @@DispSkip ; 0 = no, skip it

@@DispForce:
MOV [DispIndex],EBX ; save new task index

update the trace display

TrcIf TrcTaskSwitch
MOVZX EAX,[(DispArray.TaskPtr.Seg) + EBX*SIZE DISP_ENTRY]
CALL StrFormat,GDT_DATA,OFFSET MsgBuff,MSG_MAX, \
GDT_CONST,OFFSET cMsgDisp2, \
[DispIndex],EAX,[SwitchCtr]
CALL VidSendString,GDT_DATA,OFFSET MsgBuff
TrcEndIf

do the task switch

MOV EDX,SYNC_ADDR ; mark the start in the old task
IN AL,DX
OR AL,80h
OUT DX,AL

MOV EBX,[DispIndex] ; aim at new task pointer in table
JMP [FWORD PTR ((DispArray.TaskPtr) + EBX*SIZE DISP_ENTRY)]

IN AL,DX ; mark the end in the new task
AND AL,NOT 80h
OUT DX,AL

INC [SwitchCtr] ; count this task switch

-- return to the new task
the FAR PROC forces this to be a FAR RET...

RET
ENDP TaskDispatch

```

V86 task. Although I don't plan to use virtual memory, paging allows us to run each task in, yes, absolutely flat 3%bit protected mode. It doesn't get simpler than that, even though the behind-the-scenes code is something fierce.

And, *that* sounds like something worth knowing!

## RELEASE NOTES

The code this month builds three trivial taskettes in addition to the kernel. Each step in the process sends a torrent of trace messages to COM1—make sure you turn on your comm program's screen capture! Once everything is ready, the four taskettes run in round-robin order while the dispatcher updates a status line on the bottom of the video display.

If your system has a Firmware Development Board, you can selectively throttle the torrent of information by flipping DIP switches 15 through 11. Bit 15 controls the video status display. Watch the FDB LEDs to see the effect of turning the status off! The remaining switches control various tracing options in the task and memory routines.

Next month, we'll make the call gates more formal, add a conforming code segment, and run into a brick wall or two. Stay synced. □

*Ed Nisley, 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 74065.1363@compuserve.com.*

## SOFTWARE

Software for this article is available from the Circuit Cellar BBS and on Software On Disk for this issue. Please see the end of "ConnecTime" in this issue for downloading and ordering information.

## IRS

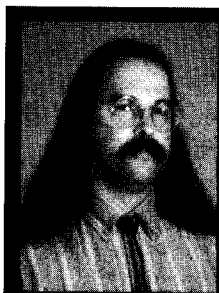
416 Very Useful  
417 Moderately Useful  
418 Not Useful

# FROM THE BENCH

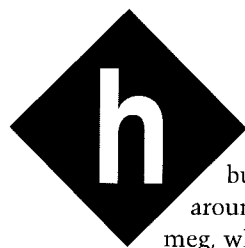
Jeff Bachiochi

## Fitting 10 oz. into a 5-oz. Package

### An Application for Highway Safety



One fax plus one modem plus one microcontroller makes one telecontroller. It's small, it's tight, and it's rugged. Jeff explains how Xecom's XE5224 is durable and independent enough to combat highway calamities.



How is it that I can buy a hard drive for around 50 cents a meg, while RAM still costs 100 times that amount? It just doesn't add up! I can pay more to fully populate a system's empty SIMM sockets than for the whole system. Don't get me wrong—I paid 50 bucks for 16 KB of RAM (that was more than I'd ever need) 20 years ago, so prices have fallen, but not in comparison to the rest of the pieces.

Overall value has certainly increased, but with every added doohickey, the MTBF goes down. Take a look at boom boxes, for instance. Today you can get a CD, double-cassette dubber, AM/FM stereo, clock, and karaoke all in one unit. What happens when one part bites the dust? (You know it will the way your kid drags it everywhere.) Nobody is gonna fix it for less than the \$100 you paid for it, so out it goes. This tendency to junk rather than fix seems true today for most appliances from telephones to steam irons.

If companies would only spend as much time on the mechanics of a product as they do on pushing the technotronics, they might last long enough for us to appreciate them. I guess it's the instant gratification that

pizazz gives and not the warranty period that most people are looking for. And so, it continues. The public gets what it wants—complexity.

I'm from the old school. I don't want fast-food audio, packaged their way. I want to hand-select each component of my audio system, mixing and matching elements to achieve a peak level of quality sound reproduction. Very few audio components these days are built from discrete parts. This new level of integration is growing thanks to MCMs (multichip modules).

#### MCM

In the mid '80s, our design group worked on expansion boards for a backplane-style microcomputer system. One such board added dual serial ports to the system—two RS-232, -422, or -485 serial ports—based on the familiar 8251 USART. While digging through some new-product announcements, an unusual component was uncovered. This module allowed some unique features to be added with little engineering effort.

The component was a small module called *MOSART*. About twice the size of an 8251 USART, the *MOSART* included an 8251, modem processor, and registered DAA. A user could add two serial ports or 1200-bps modems on the same expansion card. Since the modem ports looked like 8251s, the same software routines worked with either the stock 8251 or the *MOSART*. It was a very clean design.

Xecom is the Milipitas, California company that introduced us to hybrid modules. I like to call them "backwards." While most of us work from the processor out, designing systems toward the I/O, Xecom has worked its magic from the outside in, starting with the I/O and working back toward the processor.

#### SDRAWKCAB GNINGISED

Xecom has been making hybrids for over ten years. The line now includes higher-end, 14.4-kbps fax/data modems and all are packaged in the same DIP-style modules. Although all are worthy of elaboration, I want to

introduce you to a new level of integration—the XE5224 series module. This telecontroller incorporates a micro along with the UART, modem, and DAA into a packaged module measuring 2.75" x 1.38" x 0.5" (see Photo 1).

Although future plans include a number of different processors, the first to be released is an 8031 core. Actually, it's a Signetics 80C552 processor. The Signetics microcontroller has an S-input, IO-bit A/D converter built-in, so it adds analog-sensing capabilities to the XE5224. The 26 additional I/O bits fill out the digital portion with many of the lines providing alternate functions. These functions include real-time clock/calendar, two PWM outputs, an I<sup>2</sup>C bus, a console serial port, watchdog enable, and external interrupt and counter/timer inputs.

As most of you know, the 8031 can access 64 KB of data space and 64 KB of program space. The XE5224 includes a 32-KB SRAM for the lower portion of the data space and a 32-KB ROM (with programmed monitor and telecontroller software) for the lower portion of program space.

The upper 32-KB space is overlapped into a combined data and program space. It is composed of 32 KB of flash memory. The flash memory is actually a 128-KB device, broken into four 32-KB pages. These pages can be switched in and out of the lower 32 KB of program space and/or the upper 32 KB of data and program space. The configuration of the RAM, ROM, and flash memory is set through external pins. The variety of memory combinations this setup produces makes this module flexible in form and function.

In addition to the processor, Xecom embeds a 2400-bps modem and an optional 9600/4800 send/receive fax. The included DAA enables legal connection to the phone line for instant product acceptance.

Current consumption at 5 V for the XE5224 is less than 100 mA with the modem active and about half that with the modem in sleep mode. Although the hardware clock/calendar requires an external source (battery or

A x <CR>	analog input (O-7)	monitor displays 10-bit conversion of analog channel x
c w	change address w	monitor displays address wand allows user to change, <space> to skip, <CR> alone to end
D s e	display data-memory contents	monitor displays from address s to address e
E s e	display program-memory contents	monitor displays from address s to address e
F s e p	fill data memory with pattern	monitor fills from address s to address e with pattern p
G w	goto address w	monitor jumps to program address w and executes
H	on-line help	monitor displays these commands
I x	digital input (I/O address)	monitor displays byte read from address x
K x	kill flash section (O-3)	monitor fills 16-KB section x of active flash memory pages with FFh
L s e t	load data space from program space	monitor moves a block of program code starting with address s and ending with address e to data space starting at address t
M s e t	move data block	monitor moves a block of data starting with address s and ending with address e to data space starting at address t
O x d	digital output (I/O address)	monitor outputs byte d to I/O address x
R w	read in a hex file	monitor accepts Intel hex file and applies offset w prior to storage in the program space
T	"AT" command mode	monitor enters the modem command mode. No command prompt, # to exit back to monitor
V	version information	monitor displays version banner
W s e	write out a hex file	monitor writes an Intel hex file of the program space starting with address s and ending with address e

Table 1 -- The telecontroller comes with built-in debug and monitor commands to aid in program development

supercap), the use of nonvolatile flash memory eliminates this need for data and/or program storage.

## PRELOADED TELECOM FIRMWARE

Xecom provides a terminal emulator program for the PC which allows you to talk directly to the XE5224 through a user-supplied RS-232 interface. (Any comm program using 8N1 parameters works.) The auto-baud feature looks for a space character and returns a sign-on message indicating a link is made and the debugging monitor is active. Data rates from 300 to 38,400 bps are supported. The ">" character prompts you to input a command string. Table 1 gives a sample of the monitor commands.

Many personal computers today use multiple processors. Each processor is responsible for an individual function such as video or audio. Although most of us may not realize

it, external modems are embedded-processor based.

This is exactly the concept Xecom uses. Although it looks like an I/O device to the 80C552 processor, the modem port is actually an embedded microprocessor which shuffles parallel information and acts like a Hayes modem with the standard AT-command set (see Table 2). This takes much of the load off the '552, freeing it to do its own thing. The fax commands operate on Class 1 protocol and can be found in Table 3.

## ENVIRONMENTAL RUGGEDNESS

Neglecting the telecontroller's most obvious advantage (its size), what makes it so different from other single-board micros? I get the same feeling handling one of these modules as I do a DC/DC converter or a high-power, solid-state switch module—a sense of ruggedness and power. Potting compound encapsulates every component into a single mass. The potting

AT Commands	
AT	Command-line prefix
A	Answer incoming call immediately
AI	Reexecute last command line
B	Answer tone
Dn	Dial telephone number n; P Pulse dialing R Originate call in answer mode T Touch-tone dial ; Return to Command Mode after dialing , Pause ! 0.5-s hookflash @ Wait for silence W Wait for dial tone (default 30 s; see register S7)
E	Command Echo
H	Off-hook control
I	Display product code
L	Volume control
M	Speaker control
0	On-line control
Q	Result codes (default)
Sr?	Read and display value of register r
Sr=n	Set register r value to n
V	Word/numeric result codes
X	Response set
Z	Reset
&C	DCD control
&D	DTR control
&F	Load default
&S	DSR control
&T	Test modes
&V	Displays active user selected profiles

Table 2a—The telecontroller supports a standard Hayes A T-command set to minimize the programmer learning curve.

provides physical protection from the environment and a platform in which all components remain at a stable temperature.

When equipment has to withstand severe environmental extremes, it is prudent to design with components most likely to survive. I believe our interstate highway system would benefit from technology such as this.

## DOT/SP

The Department of Transportation in conjunction with the State Police act as a medical team to keep our asphalt arteries clean and unrestricted. If it were not for these dedicated personnel, civilization as we know it would certainly collapse. To help these professionals respond with the speed and intuition of an oracle, I wish to suggest an Auto Informer—an alternative to the present data-collection systems.

We are beginning to see the use of inductive pickups because of their simplicity. They are weather and

Digit	Code	Word Code Meaning
0	OK	Successfully executed command line
1	CONNECT	300 bps connection established
2	RING	Ring signal detected
3	NO CARRIER	Carrier not detected within Register S7 detect time
4	ERROR	Error found in command line: returns to command line
5	CONNECT 1200	1200 bps connection established
6	NO DIALTONE	No dial tone detected within 5 s after going off-hook
7	BUSY	Busy signal detected after automatically dialing a call
8	NO ANSWER	Five seconds of silence was not detected when using the @ command in the Dial command line
10	CONNECT 2400	Connection established at 2400 bps

Table 2b—Command responses also match those used by most A T-type modems, so ease code development.

traffic resistant. The sensor's signal output determines mass and even the speed of a moving body by examining the waveshapes it produces. Multiple lanes can be easily monitored using a minimum of signal conditioning hardware.

The telecontroller monitors these lanes, keeping track of the traffic's speed, relative spacing, and the numbers of cars and trucks using a specific lane. The telecontroller stores and/or reports this information with time and date stamps.

Analog inputs monitor air and road temperatures as well as humidity and precipitation. Large message signs can instantly relay driving conditions to motorists via a serial interface from the telecontroller.

Most important, the telecontroller can make telephone and fax calls to report dangerous situations like stopped traffic or icy conditions. This action may be a simple fax to the headquarters notifying them of the situation or, in a highly integrated system, the action may modem a central processor. This computer would collect and correlate information from each telecontroller. It could determine alternate routes for traffic or necessary changes to posted speed limits. Suitable messages would be relayed back to the appropriate telecontrollers by modem and displayed for the motorists.

## QUICK AS A FLASH

I wanted to experiment with the telecontroller's use of flash memory for program storage and execution. The telecontroller's ROM-based debugger occupies a

code space at 0000h with the configuration jumpers (cfg0–cfg2) set to 000. This places flash memory page 0 at 8000h.

With the monitor, I can erase the flash memory using the "K" command and download an Intel hex file into it using the "R 8000" command. The 8000 is a hex offset used to place an object file (created to run at 0000h) into the flash memory (page 0) at 8000h. When the cfg0 jumper is changed, the flash memory (page 0) is relocated to 0000h, replacing the monitor. The program in memory executes automatically after reset (or powerup).

In addition to being relocated, the flash memory is now write protected, which makes development a breeze. There are no EPROMs to burn. Hey! Wait a minute. Did I just suggest that writing a program in assembler or C was a breeze?

If you follow this column on a regular basis, then you know I can't stand spending hours and hours at a terminal writing code. And, when I can get away with it, I like to use BASIC (I understand a future version of the telecontroller may be available with embedded BASIC). In addition, the user has at his disposal a multitude

Command	Description
+FCLASS=n	Select fax class
+FTS=n	Stop transmission and wait
+FRS=n	Receive silence
+FTM=n	Transmit data
+FRM=n	Receive data
+FTH=n	Transmit data with HDLC framing
+FRH=n	Receive data with HDLC framing

Table 3—The Class I command set offers control of the various fax features.



of C library routines. However, to put this hardware through its paces today, I am forced to get down to the bitty gritty. Well, almost.

A few years ago, I did a column on magnetic levitation and used a BASIC compiler for speed-speed in development and speed in execution. It's time to pull this back off the shelf and try integrating it with the 80C552 to create an executable file.

Since the 80C552 is an offshoot of the 8031 core, I can compile a BASIC program and execute it as an 8031. However, I need some assembly routines for accessing any on-chip (80C552) functions that I might wish to take advantage of (e.g., the 10-bit

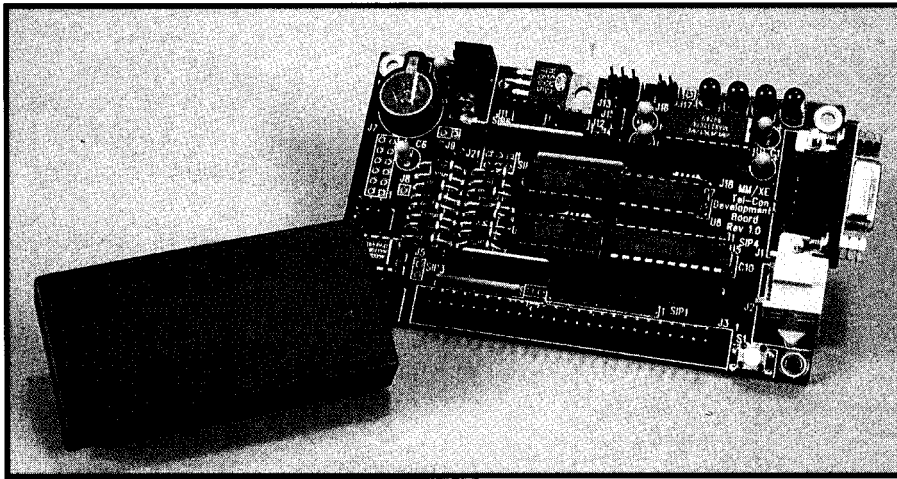


Photo 1—The XE5224 telecontroller's small size (2.75" x 1.38") influenced the development board's maximum allowable dimension of 4.30" x 2.60".

A/D converter). The BASIC compiler allows in-line assembly code to be integrated right along with the BASIC, so it isn't entirely a hodgepodge.

#### MONITORING OVERVIEW

In our highway application, speed and volume monitoring is based on timings calculated from dual, buried mass sensors installed in each lane of

the highway's road bed. (I think this would be a great project for fuzzy logic using only single lane sensors, but that's not my intent—at least not in this article.)

Using dual sensors allows for easy speed and size detection. The time difference between tripping the first

sensor and the second sensor gives the speed of the passing mass. If the sensors are offset by four feet, a vehicle travelling at 60 mph passes the sensor's trip points in 44 ms.

The length of time the mass spends over the first sensor in comparison to its speed determines the size of the mass. In gross terms, if the same vehicle remains over the first

## REMOTE POWER CARD!

3 VERSIONS:

### RESET

WATCHDOG RESETS PC IF IT HANGS FOR HARDWARE OR SOFTWARE REASONS

### PHONE

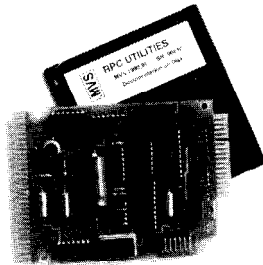
TURN ON PC WITH PHONE, SHARE VOICE / MODEM LINE, CONTROL AC APPLIANCES

### TIMER

WAKEUP OR SHUTDOWN PC, LATE NITE BACKUP / MODEM, CONTROL AC APPLIANCES

**95\$** 27\$ OEM

ALL MVS CARDS INCLUDE ASM, BASIC, AND C SOURCE FOR PC OR SBC



## 8 CHAN ADC

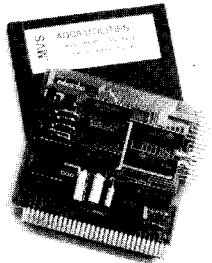
DATA ACQUISITION, SERVO CTL, AUDIO  
8-BIT RESOLUTION 22KHZ SAMPLE RATE  
SHARP CUTOFF ANTI-ALIAS FILTER  
CREATE STEREO BLASTER (.VOC) FILES

**95\$**

## 2 CHAN DAC

VOICE MAIL, MUSIC, ALARMS, CTLVOLT  
8-BIT RESOLUTION 44KHZ SAMPLE RATE  
PLAYS MONO / STEREO BLASTER FILES  
FUNCTIONS AS DIGITAL ATTENUATOR TOO

**75\$**

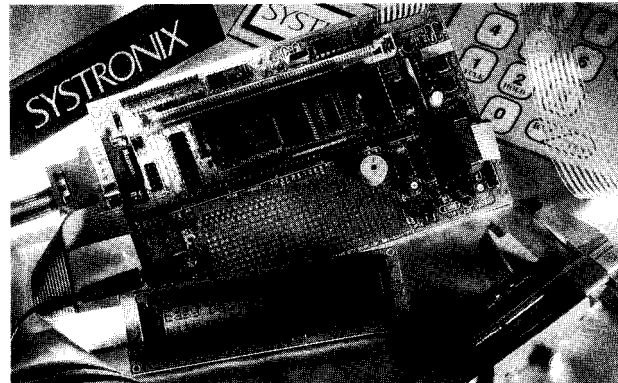


**MVS**

MVS BOX 850  
MERRIMACK, NH  
(508) 782-9507

5 YEAR LIMITED WARRANTY  
PING IN USA

## NEW! UNIVERSAL DALLAS DEVELOPMENT SYSTEM from \$199!



- It's a complete 8051-family single board computer!
- One board accommodates any 40 DIP DS5000, 40 SIMM DS2250, 40 SIMM DS2252, or 72 SIMM DS2251, 8051 superset processor! Snap one out, snap another in.
- Programs via PC serial port. Program lock & encrypt.
- LCD interface, keypad decoder, RS232 serial port, 8-bit ADC, four relay driver outputs, four buffered inputs.
- Power with 5VDC regulated or 6-13 VDC unregulated
- Large prototyping area, processor pins routed to headers
- Optional enclosures, keypads, LCDs, everything you need
- BC151 Pro BASIC Compiler w/50+ Dallas keywords \$399

**SYSTRONIX®**

TEL: 801.534.1017 FAX: 801.534.1019  
555 South 300 East, Salt Lake City, UT, USA 84111

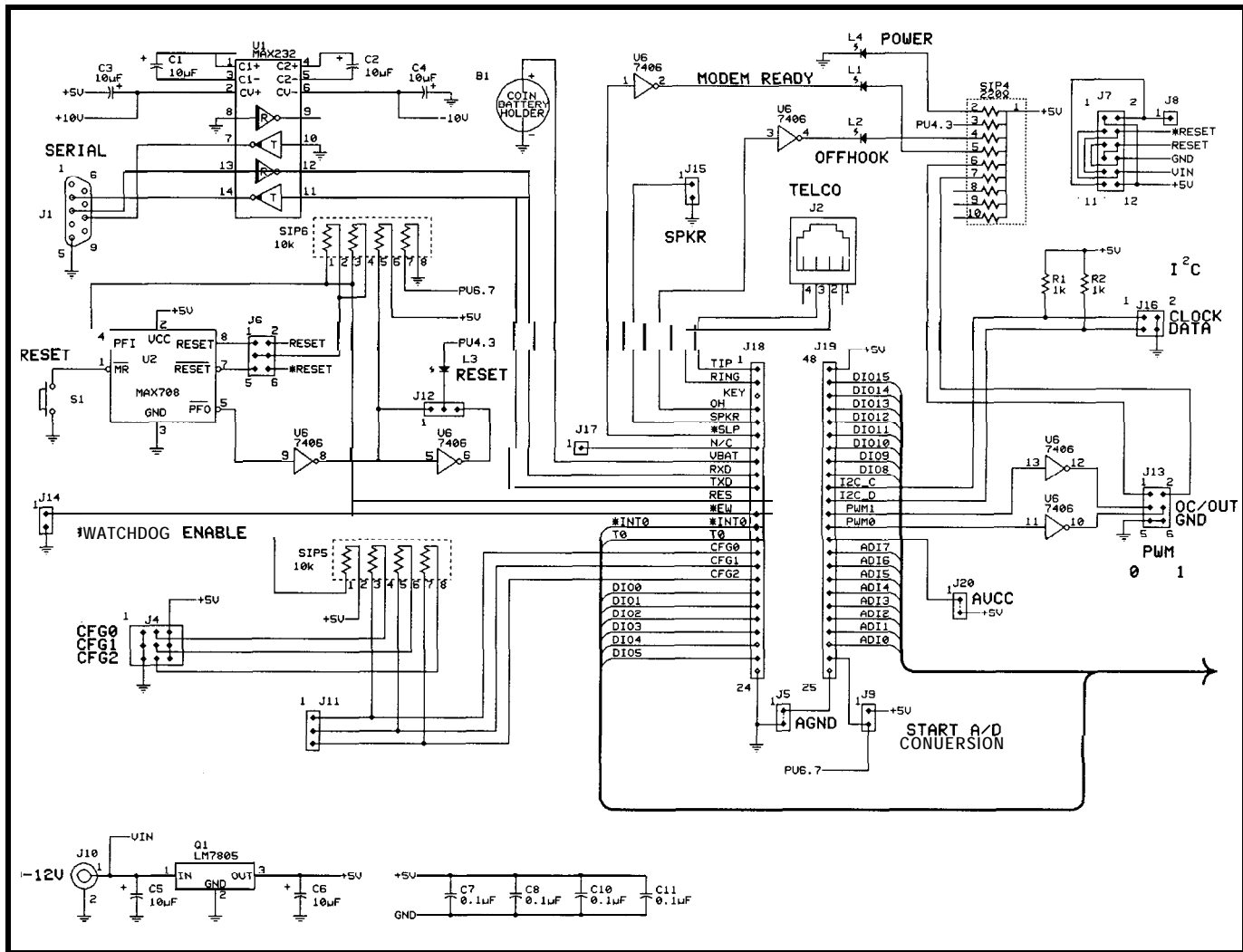


Figure 1 a--The development board adds support circuitry which prevents accidental module destruction and provides easy access to all XE5224 functions.

sensor for 132 ms, it is three time units by four feet (i.e., 132 ms/44 ms x 4') or twelve feet. The high, low, and average speeds are saved for each lane. The volume is broken down into three groups: autos and motorcycles are recorded as small vehicles, single-box trailers and RVs as medium vehicles, and double-box trailers and some hauled, heavy-construction equipment as large vehicles.

Monitoring the air and road-bed temperature is extremely important in this application. Your choice of sensor determines what (if any) sensor calibration may be necessary. Silicon temperature sensors output 10-mV per degree (negative output for negative temperatures). By adjusting the offset and gain of the incoming signal to take full advantage of the A/D converter's 0-5-V input, it's possible to get 0.5" resolution.

### PUTTING IT ALL TOGETHER

Although I wanted to use the optional fax modem for sending recorded data, I did not want to go through the hassle of breaking down the data characters into line-scan bits. The line-scan bits, of course, have to be converted into the modified Huffman code necessary for facsimile transmissions. Ugh, another potential article.

This time, I'll be satisfied with simple 2400-bps data modem transmissions. The telecontroller makes a call to the central computer at hourly intervals or whenever a dangerous situation occurs. The central office can also call the telecontroller to ask for the latest data.

The modem's AT command set (Table 2) allows the micro easy access. The most difficult part of modem communications is parsing the output

command strings since they are a mix of text and values. (An output driver accessed through BASIC's UO1 command would make this operation a breeze.)

### WRAP UP

All told, I'm really impressed with this little package. There is an established core audience for this 8031-family device. The digital and 10-bit analog I/O are ample for most applications. The inclusion of the hardware clock/calendar and integrated modem complements the processor's power rather nicely.

I liked this module so much I designed a small 2.5" x 4" interface board (see Figure 1) to help in the early stages of software and hardware development. I know before you develop a piece of hardware that uses the telecontroller, you will want to

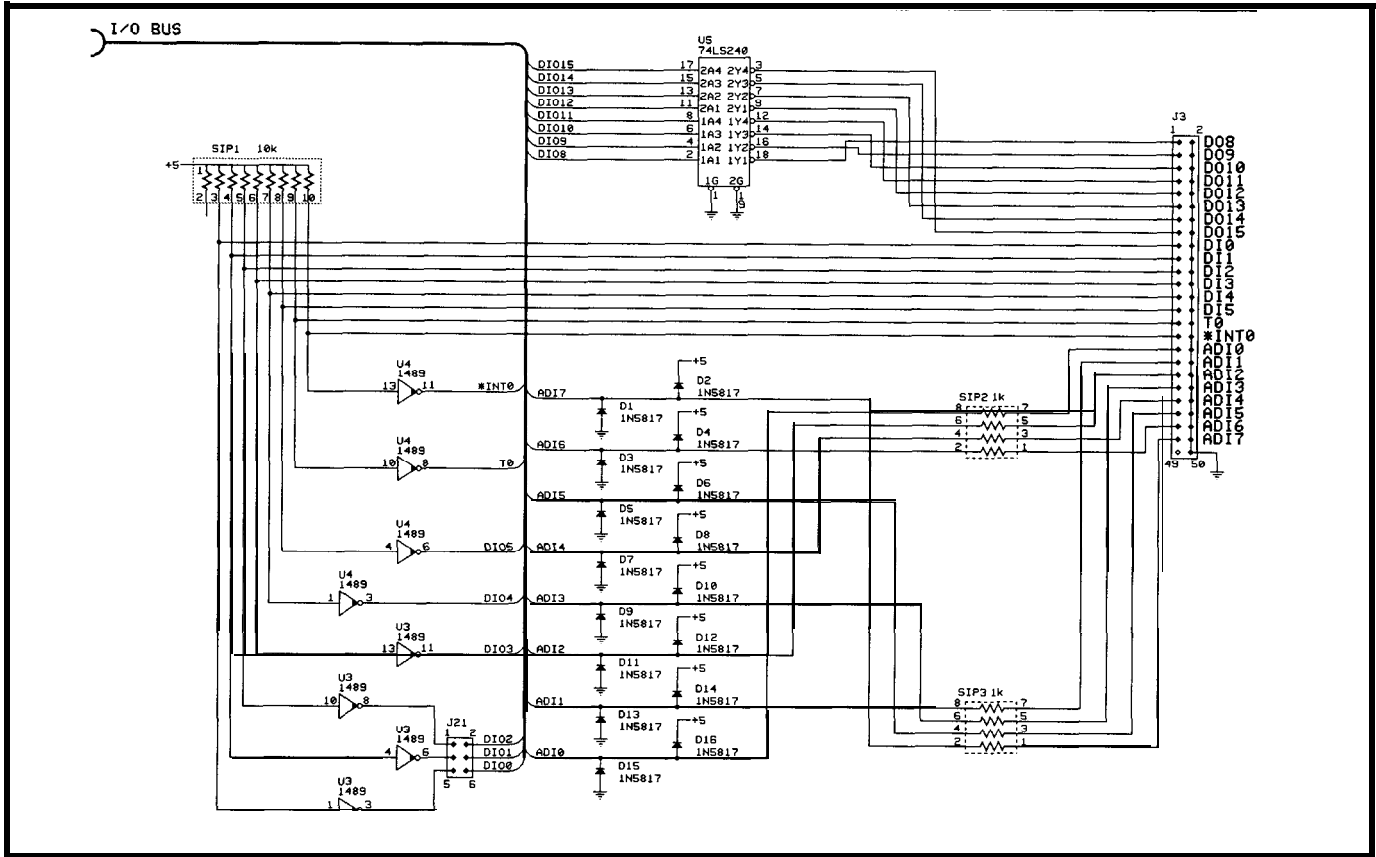


Figure 1 b—The digital I/O is brought out to an industry-standard 50-pin header to allow the use of external I/O backplanes.

WORLD'S SMALLEST

# 486<sup>TM</sup> DX

Embedded PC with  
Floating Point,  
Ethernet & Super  
VGA Only 4" x 4"

The PC/II +i includes:

- 486DX<sup>TM</sup> CPU at 25MHz or 33MHz clock frequency
- Full 8K Cache with Floating Point
- Ethernet local Area Network
- Local Bus Super VGA Video/LCD
- Up to 2MBytesFlash<sup>TM</sup> with TFSS
- 4 or 16MBytes User DRAM
- PC/I 04 or ISA Bus compatible option (with adapter)
- 4" x 4" Format; 6 watts power consumption at 15 volt only

486DX and Flash are registered trademarks of Intel Corp. as are PC, AT of IBM, Megatel of Megatel Computer (1996) Corp.

(416) 245-2953

megatel®

125 Wendell Ave. • Weston, Ont. • M9N 3K9 • Fax: (416) 245-6505

## MOVE OVER INTEL MICROMINT SOURCES 80C52 CMOS BASIC CHIP

Micromint has a more efficient software-compatible successor to the power-hungry Intel 8052AH-BASIC chip. The 80C52-BASIC chip was designed for industrial use and operates beyond the limits of standard commercial-grade chips. Micromint's 80C52-BASIC chip is guaranteed to operate flawlessly at DC to 12 MHz over the entire industrial temperature range (-40°C to +85°C). Available in 40-pin DIP or PLCC

80C52-BASIC chip	\$19.00
OEM 100-Qty. Price	\$12.00
BASIC-52 Prog. manual	\$15.00

MICROMINT, INC.  
4 PARK ST. • VERNON, CT 06066  
TO ORDER CALL  
**1-800-635-3355**

play a bit with it. Well, this will get you started at full throttle.

The DE9 connector and RS-232 interface enable developers to make an easy console connection. The onboard 5-V regulator and power jack make stringent regulated-power-supply requirements unnecessary.

Xecom hints at using other micro cores in the future. To prepare for that possibility, I chose to use a MAX708 hardware reset. This ensures suitable reset timing and both polarities of reset for future products. Four onboard LEDs indicate status of reset, power, modem ready, and off-hook conditions.

To reduce the potential damage to the telecontroller, I added protection to both the TTL and analog I/O. During the development stages, TTL I/O is protected by buffering the inputs for high-voltage  $\pm 30 V_{IN}$ . The outputs are buffered or inverted by using either a '240 or a '241. Analog inputs are protected by 1-k $\Omega$  resistors and Schottky diodes tied to the power-supply rails. All I/O is available on a 50-pin header that is compatible with

industry-standard I/O racks. Telco connections are through a standard RJ-11 connector.

All together, it's a pretty nifty combination.

*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

The following are available from:

Xecom, Inc.  
374 Turquoise St.  
Milipitas, CA  
(408) 9456640  
Fax: (408) 942-1346

Micomint, Inc.  
4 Park St.  
Vernon, CT 06066  
(203) 871-6170  
Fax: (203) 872-2204

**XE5200 (Blackjack 552)**  
(controller only) \$ 1 3 9  
**XE5224DM (Blackjack 552DM)**  
**Telecontroller** (with data  
modem) ..... \$199  
**XE5224FD (Blackjack 552FD)**  
**Telecontroller** (with fax and  
data modem) Call for pricing  
**Blackjack Protective Demo**  
Board..... \$99

The XE5200 family is available second sourced by Micromint under the tradename Blackjack 552.

BC15 1 BASIC Cross-Compiler  
(integer math)  
Systronix, Inc.  
754 East Roosevelt Ave.  
P.O. Box 526398  
Salt Lake City, UT 84152  
(801) 487-7412

## IRS

419 Very Useful  
420 Moderately Useful  
421 Not Useful

## ► Good Stuff ◀

Bar Code Sensor  
Battery Controllers  
Clock/Calendars  
Digital Power Drivers  
DTMF & Phone Interfaces  
Firmware Furnace Widgets  
HCS-II Hard-to-find Parts  
I<sup>2</sup>C Bus ICs  
IRLEDs & Photodiodes  
IR Data Link Parts  
IR Remote Control  
Laser Diode Controllers  
Linear Hall Effect Sensor  
Multiplexers & Crosspoints  
Power Op Amp  
Remote Temperature Sensor  
Stepper Motor Drivers  
Watchdogs & Power Monitors  
8051 Information  
and more!

Use a soldering iron? Get the parts!

UPS. Ground/2nd day \$6 5019.00 to 48 US states. COD add \$4.50 PO Boxes and Canadian addresses: \$6 for USPS mail. Check, MO, or COD only; no credit cards, no open POs. NC residents add 6% sales tax. Quantity discounts start at five parts. Data sheets included with all parts.

Call/write/fax for serious/ly tempting catalog...

### Pure Unobtainium

► Your unusual part5 source ◀

13109 Old Creedmoor Road Raleigh NC 27613-7421  
FAX/voice (919) 676-4525

## BCC52 BASIC-52 COMPUTER/CONTROLLER

The BCC52 controller continues to be Micromint's best selling single-board computer. Its cost-effective architecture needs only a power supply and terminal to become a complete development system or single-board solution in an end-use system. The BCC52 is programmable in BASIC-52, (a fast, full floating point interpreted BASIC), or assembly language.

The BCC52 contains five RAM/ROM sockets, an "intelligent" 27641128 EPROM programmer, three 8-bit parallel ports, an auto-baud rate detect serial console port, a serial printer port, and much more.

### PROCESSOR

- 80C528-bit CMOS processor w/BASIC-52
- Three 16-bit counter/timers
- Six interrupts
- Much more!

### Input/Output

- Console RS232 autobaud detect
- Line printer RS-232
- Three 8-bit parallel ports
- EXPANDABLE!
- Compatible with 12 BCC expansion boards

### MEMORY

- 48K RAM/ROM, expandable
- Five on-board memory sockets
- Either 8K or 16K EPROM

To Order Call 1-800-635-3355

Tel: (203) 871-6170

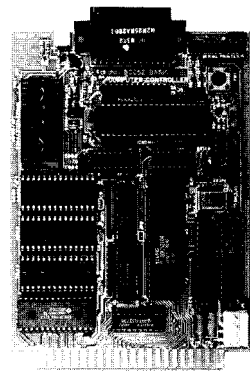
Fax: (203) 872-2204

BCC52	Controller board with BASIC-52 and 8K RAM	\$189.00	Single Qty.
BCC52C	Low-power CMOS version of the BCC52	\$199.00	
BCC52I	-40°C to +85°C industrial temperature version	\$294.00	
BCC52CX	Low-power CMOS, expanded BCC52 w/32K RAM	\$259.00	

CALL FOR OEM PRICING



**MICROMINT, INC.** 4 Park Street, Vernon, CT 06066  
in Europe (44)0285-658122 • in Canada, (514) 336-9426 • in Australia: (3)467-7194  
Distributor Inquiries Welcome!



# I Sync, Therefore I DRAM

## SILICON UPDATE

Tom Cantrell

These days, Intel is mainly known as the company that, in concert with Microsoft, controls what goes on inside most PCs. Their commanding position has served them well, propelling them to the top of the IC business.

After a decade of PC and x86 hoopla, it's easy to forget Intel's earlier contributions at the dawn of the silicon revolution. Long before the x86, Intel invented a chip that is arguably even more important—the DRAM.

Looking back at chips like the early '70s' Intel 1103 (see Figure 1a) is always good for a laugh. Imagine what your typical 1-MB PC would be like if it had 8000 of these little puppies. Forget the mini- and maxi-tower enclosures; the Trump tower is more like it. You'd better beef up your power supply too since each 1103 burned a half watt or so. Roll-your-own precharge, inverted DOUT, and 16-V I/O levels just add to the fun.

Nevertheless, as feeble as it now seems, the 1103 started the DRAM

ball rolling down a slippery slope of ever-increasing density and (at least until recently) ever-falling prices. Now, with the DRAM market moving through 4 and 16 Mb to beyond, a few chips are enough to handle the most bloated software upgrades without blinking a bit.

The modern DRAM era began in the late '70s with a 16-Kb DRAM such as the Intel 2116 illustrated in Figure 1b. More important than the technical features, acceptance of a standard kicked off multisourced DRAM as a commodity. The subsequent brutal battles for sockets had great benefits for users, but drove many suppliers (including Intel) from the business.

### DRAM 01

Though the 2116 came (and went) before the x86, most of its features persist to this day. Thankfully, a single supply replaces the three (+5, +12, and -5 V) of yesteryear, but otherwise DRAMs haven't changed that much.

A typical  $2^n$ -bit DRAM is organized as  $2^{n/2}$  rows and  $2^{n/2}$  columns. As shown in Figure 2a, the row address is strobed into the DRAM with RAS\* [row-address strobe]. Next, the column address is strobed with CAS\* [column-address strobe], which also clocks the data in or out depending on the WE\* pin. The timing parameter  $t_{RAC}$  denotes the access time of the DRAM (i.e.,  $t_{RAC}$  is 70 ns for a "70-ns DRAM").

To refresh your memory (ho-ho), DRAMs store each bit on a capacitor that leaks. Thus, they must be periodically refreshed, usually a row or two (limited by power considerations) at a time. Typical specs call for the entire



Tom reminds us that in the beginning,

DRAM was created. And, after many days, Hitachi brought forth synchronous DRAM, which offers better timing, greater speed, and more bandwidth. Tom says that this is good.

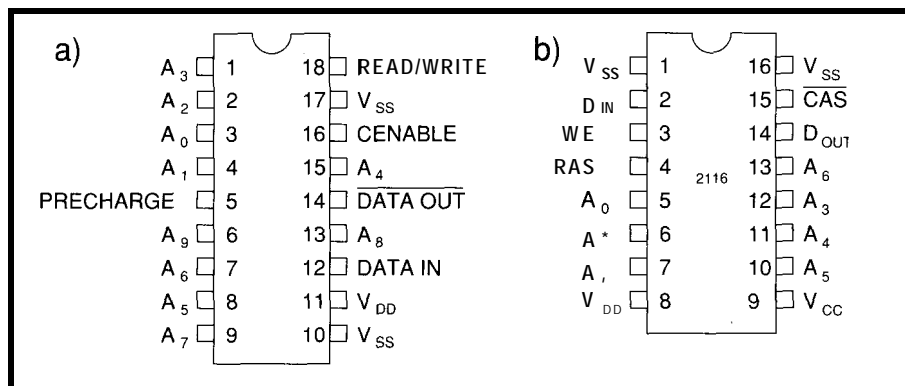
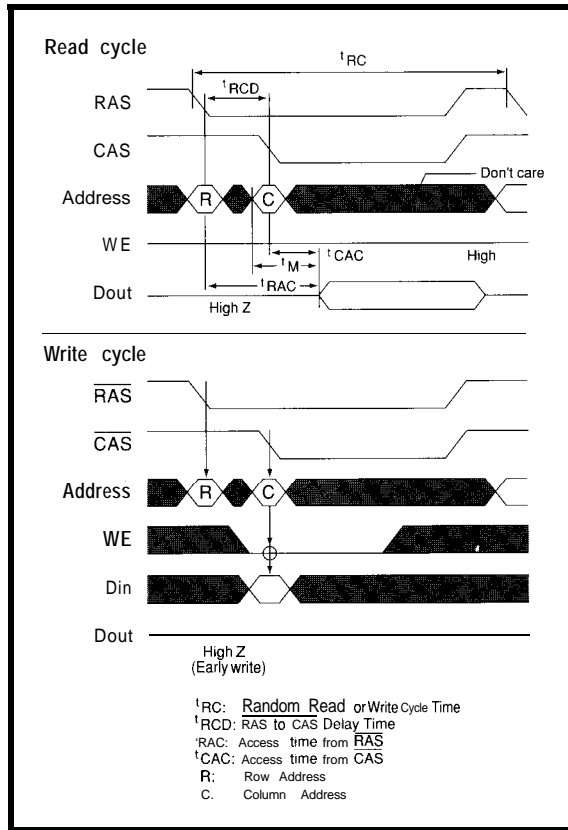


Figure 1-a) The Intel 1-Kb 1103 DRAM was one of the first DRAMs. b) The modern DRAM era started with 16-Kb DRAMs like the 2116.

Figure 2a—Basic DRAM read and write cycles use a RAS\*-before-CAS\* technique. The  $t_{RAC,spec}$  determines the access speed of the DRAM.



DRAM to be refreshed hundreds of times per second, though those pesky capacitors often prove surprisingly more bitworthy than that.

On average, a refresh cycle needs to occur every 15 us or so (i.e., 1-Mb DRAM needs 5 12 refresh cycles every 8 ms, a 4-Mb DRAM requires 1024 cycles every 16 ms, etc.). But, there's nothing that says you can't group all your refresh cycles in a burst.

Figure 2b shows the popular types of refresh cycles. The first is simply to read a location in each row, a scheme well-suited to graphics applications or software-refresh schemes. The similar RAS\*-only refresh cycle was the most widely used on the 16-, 64-, and 256-Kb DRAMs. Starting with 1-Mb DRAMs, CAS\*-before-RAS\* (also known as CBR) refresh was offered and has since become popular. With an integrated refresh-address counter, CBR doesn't require an external address.

About the same time as the 1-Mb DRAMs came into existence, the formerly simple-minded DRAMs started to get brainy with the addition of high-speed access modes (shown in Figure 2c). All modes rely on the fact that once a row is accessed, it's possible to access different columns quickly, without requiring a new RAS\* cycle.

Page mode repeats only the CAS\* cycle for each new column address, cutting access and cycle time nearly in half. Note that the column address need not be sequential. Nybble mode latches four columns for subsequent access by CAS\*. Since access is always sequential, no column address is required, making nybble mode even

faster than page mode. Static-column mode, as the name implies, offers speedy SRAM-like access to the column depending solely on the address, not CAS\*.

Though a little slower than the others, page mode has proven the market favorite. Recently, a variant of page mode called *Extended Data Out* (EDO) has been introduced, which quickens page-mode cycles by leaving the output driver on between CAS\* pulses. It looks like EDO is going to be widely adopted, further cementing page mode's position as the high-speed access mode of choice.

Subsequently, DRAM technology has migrated into a variety of specialized ICs including video

RAM, TV line memory, pseudo-SRAM (PSRAM), and so on. Nevertheless, the plain old, x1 or x4 page-mode DRAM reigns supreme in the marketplace.

Now, fueled by the insatiable demand for more bandwidth (for less \$, of course), a whole raft of clever new DRAMs are crawling out of the lab. One particular model appears headed for success—thanks not only to good features, but also to the warm, fuzzy glow multisource standardization imparts. Say "Hello" to the new kid on the block, the *synchronous DRAM*, otherwise known as SDRAM.

### ROCK 'EM, CLOCK 'EM

JEDEC (Joint Electron Device Engineering Council), a formerly mild-mannered international standards committee, has emerged as a powerful force in the memory market.

Originally, they, like many standards organizations, reacted to market realities by documenting after-the-fact standards. However, as the demand for standardization has grown, their mediation and blessing has become very important. JEDEC played a major role in making commodity SDRAMs a reality by forging a peace agreement between headstrong contenders.

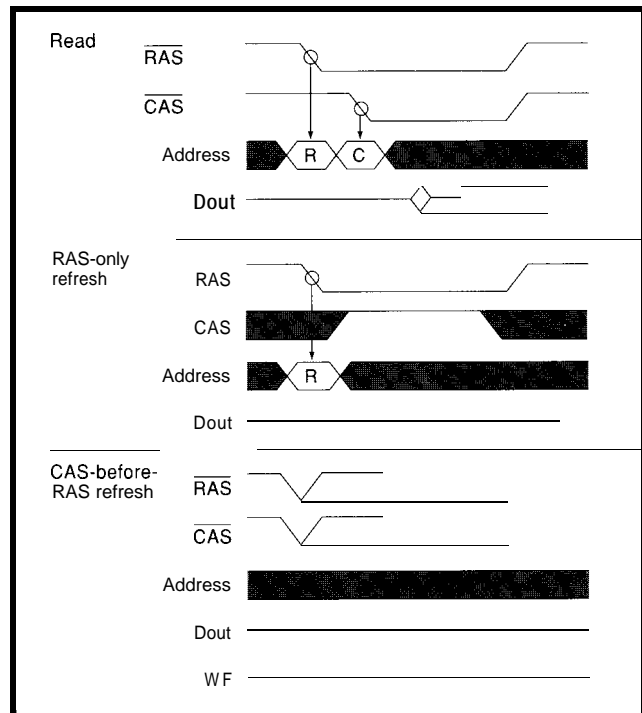
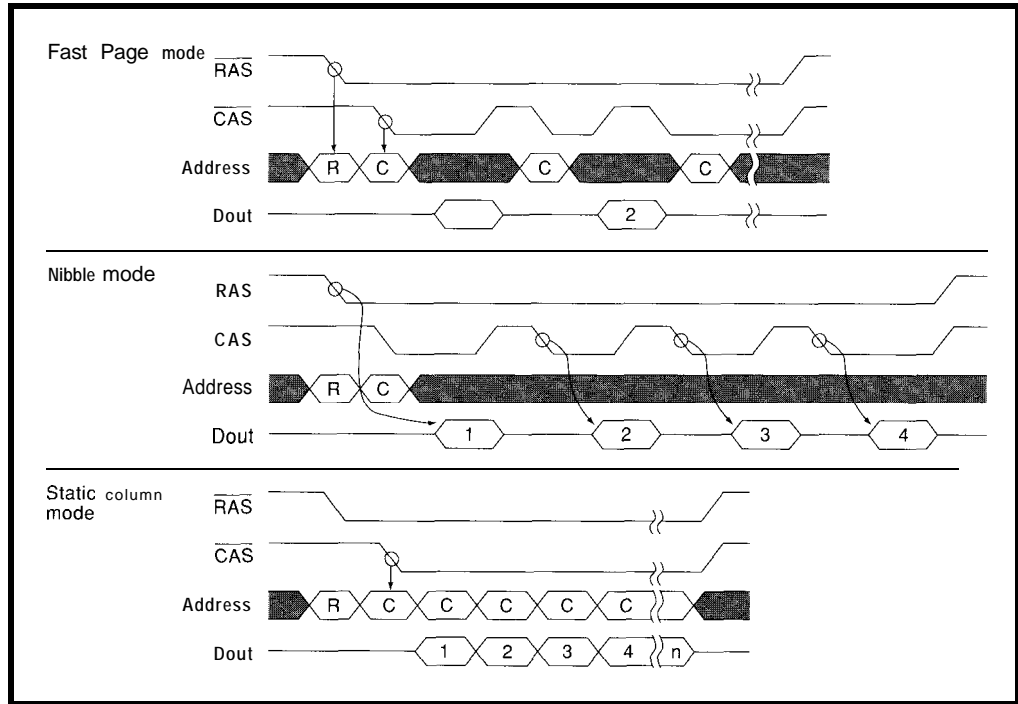


Figure 2b—The Read and RAS\*-only refresh methods, which require an address, are being supplanted by CAS\*-before-RAS\*, which doesn't.

Figure 2c—Of the three high-speed access modes, Fast-Page mode is most widely used.



Let's start by taking a look at one of the first SDRAMs out of the blocks, the 4-Mb, 66-MHz Hitachi HM5241605 (see Figure 3). Thanks to JEDEC standardization, most of the discussion applies to SDRAMs from other suppliers, though you've always got to be on the lookout for minor spec differences.

The first thing you'll notice is it's wide (x16), thankfully easing the frustration caused by the narrow xl and x4 DRAMs. So far, it looks like SDRAMs (4 Mb and 16 Mb) will initially be offered in x4, x8, x9, x16, and x18 configurations. Later, I expect we'll see (64 Mb) x32.

SDRAMs reflect the modern downsizing trend in their use of TSOP (Thin Small Outline Package) surface-mount technology. Thanks to TSOP, the miniaturization possibilities are truly impressive—% MB (2 MB for 16-Mb SDRAM) fits in a footprint about the same as the old DIP ( $\approx 10 \times 20$  mm) and just a little thicker (1.2 mm) than a business card.

The easiest way to move bits faster with less power is to move them less far, so SDRAMs downsize the power supply to 3.3 V  $\pm 10\%$ . Now here's a place you need to check an individual SDRAM data sheet. It looks like some SDRAMs may achieve 5-V TTL compatibility by tweaking their AC specs. However, this capability may disappear in future-generation SDRAMs with

finer geometries. In any case, for the same speed and power reasons, whatever the SDRAM connects to is increasingly likely to be 3.3 V too.

Otherwise, the pin names certainly look familiar. There's RAS\*, CAS\*, WE\*, CE\*, and multiplexed addresses just like before. The data lines (I/O0–I/O 15) are bidirectional, but the separate DIN and DOUT lines of the plain DRAMs were usually

connected together on the PCB anyway. The SDRAM features separate power for the memory array ( $V_{CC}$  and  $V_{CCQ}$ ) and I/O pins ( $V_{CCQ}$  and  $V_{SSQ}$ ) to help isolate noise.

DQMU and DQML are individual byte enables for the data bus, necessary to support an atomic, byte-write operation. Otherwise, a byte write would require a word read, byte mask, word write sequence.

It's the clock pin (CLK) and its enable (CKE) that are most important. After all, that's what the S in SDRAM is all about. While a plain DRAM is kind of passive, sitting around waiting for someone to tickle RAS\* and CAS\*, an SDRAM is a rather nervous critter that wants to blast some data on every clock edge (see Figure 4). With clock rates up to 66 MHz available now and with 100 MHz just around the corner, an SDRAM can handle the data rates demanded by superduper CPUs and gee-whiz graphics.

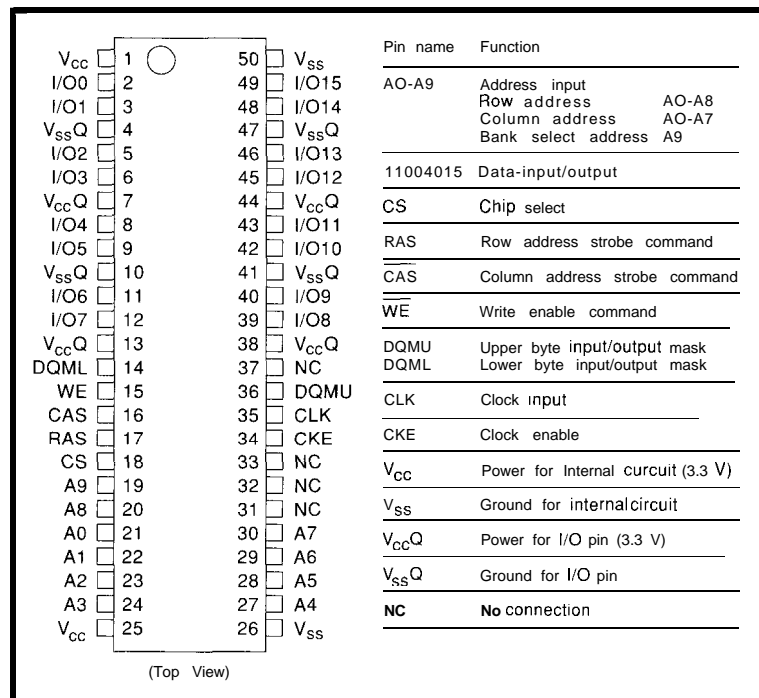


Figure 3—The Hitachi HM5241605 synchronous DRAM might be considered a New Age DRAM.

## ELECTRONIC BANKING

Looking inside the SDRAM (Figure 5) reveals differences hidden behind the familiar-sounding pin names. The most obvious difference is that the memory array is split into two banks (selected using A9). As we'll see later, this supports a ping-pong, bank-switching scheme which maximizes bandwidth.

The other thing to note is that the familiar-sounding pin names—RAS\*, CAS\*, WE\*, and so on—no longer actually connect to latches and buffers, but instead feed a control-logic block.

It turns out these pins don't actually control the memory transfer (remember, CLK does that), but should be interpreted as bits in an opcode as shown in Figure 6. The pins are sampled at each CLK cycle and inject a command into the SDRAM pipeline unless in a DESL (CS . high) or NOP (RAS\*, CAS\*, WE\* high) state. The first command to get familiar with is MRS (Mode Register Set) since it defines the key SDRAM operating characteristics (see Figure 7).

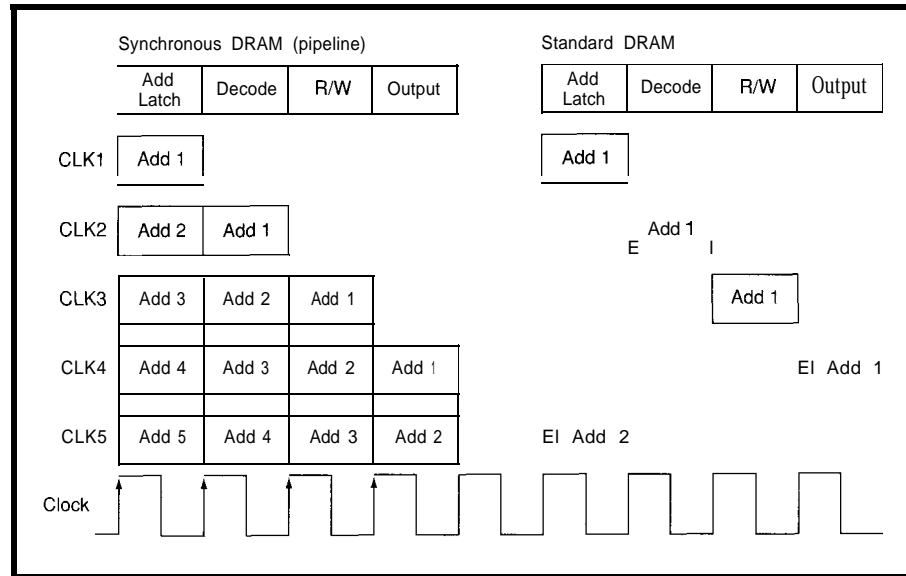


Figure 4—Synchronous DRAMs rely on a high-speed clock (e.g., 66 or 100 MHz) to time all commands and data transfers.

*Burst length* refers to how many data transfers and clocks are associated with each READ or WRITE command, the choices being 1, 2, 4, 8, and full page (which, in this case, is 256). Shorter bursts end automatically, while the *BST* command terminates full-page

bursts. *Burst type* refers to the addressing order of the burst and is normally sequential, unless you're connected to a '486 or Pentium, which feature a unique(!) interleaved, cache line fill.

*Write mode* offers the option of burst read and write or burst read with

*Find out how you can add intelligence to any home, at a cost that's within your budget.*

**LIVING WITH AN INTELLIGENT HOME**  
*will change the way you live.*

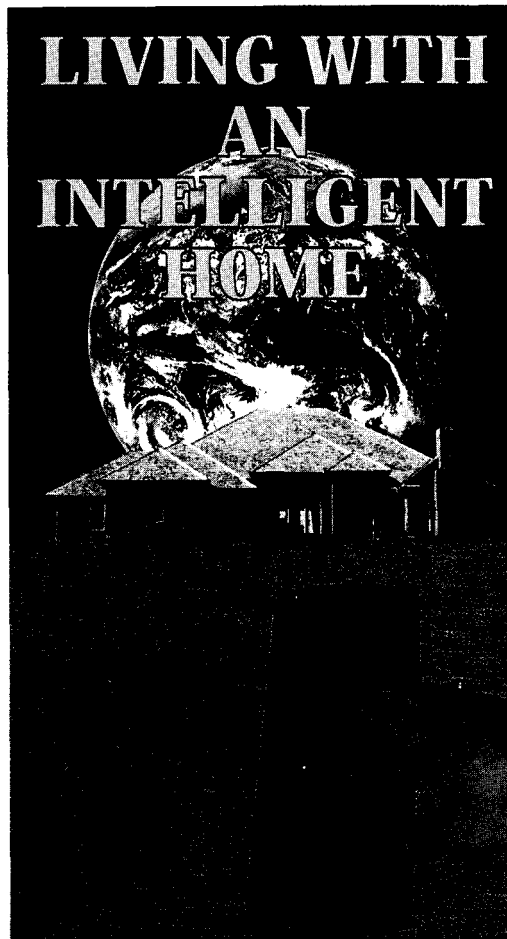
*Written by David Gaddis, author of Understanding & Installing Home Systems*

*This VHS video cassette retails for \$24.95 plus \$5 s&h.*

*It is being offered to Circuit Cellar INK subscribers for only \$17.95 plus \$4 s&h (U.S.)*

**ORDER TODAY!**

*Don't let this exciting technology opportunity pass you by!*



*Take a tour through modern technology in today's home...*

**Circuit Cellar, Inc.**

4 Park St. • Vernon, CT 06066

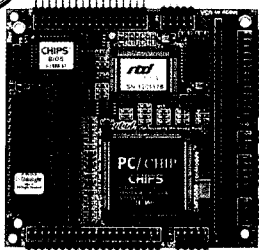
Tel: (203) 8752752

Fax (203) 872-2204



# Replace Four Conventional PC/104 Modules with One SuperXT™ CMF8680 cpuModule™

Embedded PC/XT Controller with Intelligent Power Management



PC/104 Compliant  
Actual Size: 3.6 x 3.8 x 0.6"

**\$449**  
100 pcs

- PC/XT compatibility with 286 emulation
- 14 MHz, 16-bit 8086 CPU
- +5V only; 1.6W at 14.3 MHz, 1W at 7.2 MHz
- Intelligent sleep modes, 0.1 W in Suspend
- ROM-DOS and RTD enhanced BIOS
- Compatible with MS-DOS & real-time operating systems
- 1M bootable Solid State Disk & free software
- 4K-bit configuration EEPROM (2K for user)
- 2M on-board DRAM
- IDE & floppy interfaces
- CGA CRT/LCD controller
- Two RS-232 ports, one RS-485 port
- Parallel, XT keyboard & speaker ports
- Optional X-Y keypad scanning/PCMCIA interface
- Watchdog timer & real-time clock

Expand This Or Any PC/104 System with the

## CM106 Super VGA Controller utilityModule™

- Mono/color STN & TFT flat panel support
- Simultaneous CRT & LCD operation
- Resolution to 1024 x 768 pixels
- Displays up to 256 colors

**\$223**  
100 pcs

## Speed Product Development with the DS8680 Development System

Your DS8680 includes the CMF8680, CM102 keypad scanning/PCMCIA, CM104 with 1.8" 85MB hard drive, CM106 SVGA controller & DM5406 12-bit, 100 kHz dataModule™ in an enclosure with external power supply, 3.5" floppy, keyboard, keypad, TB50 terminal board, SIGNAL\*VIEW™, SIGNAL\*MATH™, MS-DOS, SSD software & rtdLinux™ for just

**\$2950.**

For more information on our PC/104 and ISA bus products, call today.



**Real Time Devices USA**

200 Innovation Blvd. • P.O. Box 906  
State College, PA 16803 USA  
(814) 234-8087 / Fax: (814) 234-5218

**RTD Europa • RTD Scandinavia**

Real Time Devices is a founder of the PC/104 Consortium

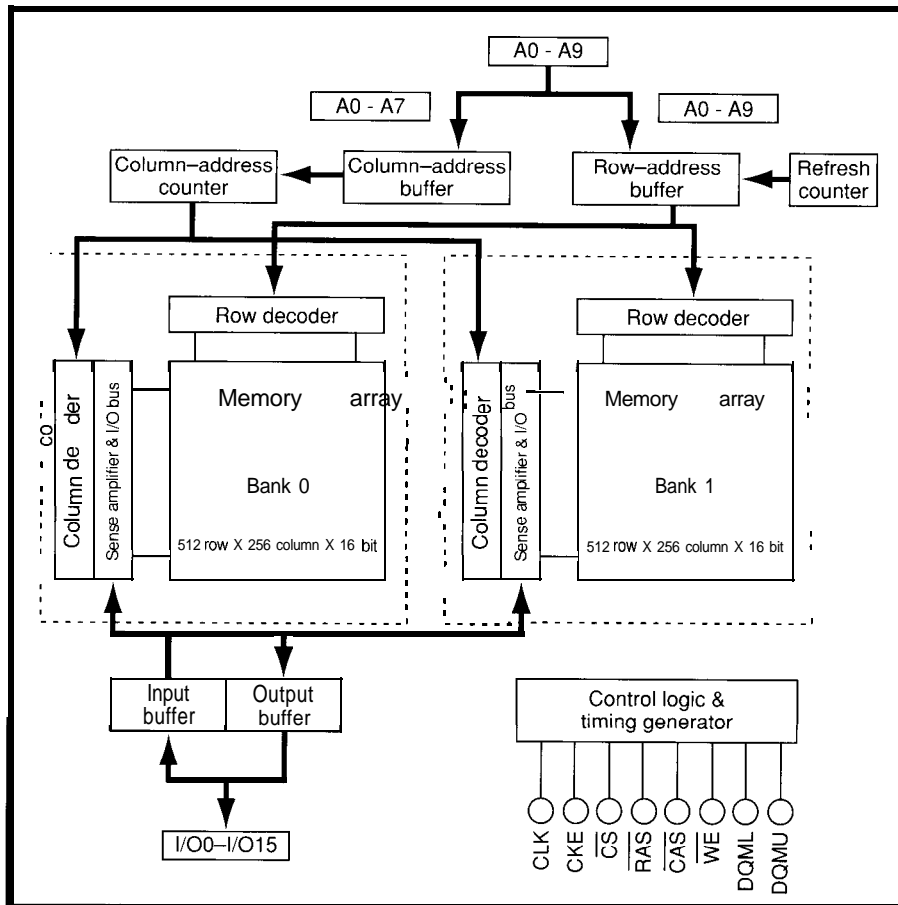


Figure 5—Inside, synchronous DRAMs are split into two banks for maximum performance. Many of the control pins have familiar names, but now, they are little more than the opcodes for an on-chip logic that controls the actual memory access.

single write. The latter mode may be useful in graphics applications (e.g., a burst read refreshing the CRT and single writes coming from the drawing processor) or to support a CPU with write-through cache.

CAS\* latency depends on how many clock cycles elapse between a request and transfer, with higher clock

rates demanding more delay. For instance, a particular-speed SDRAM may achieve 15-ns clock cycle and access times with CAS\* latency 2, but only 30 ns with CAS\* latency 1. Figure 8 summarizes the CAS\* latency and burst-length options.

As shown, a transfer is initiated with an ACTV (activate) command,

Function	Symbol	CKE	CS				RAS		CAS		WE		A9	A8	A7	A0
		n	n	n	n	n	n	n	n	n	n	n	n	n	n	n
Ignore command	DESL	H	X	H	X	X	x	x	x	x	x	x	x	x	x	x
No operation	NOP	H	X	L	H	H	H	H	H	X	X	X	X	X	X	X
Burst stop in full page	BST	H	X	L	H	H	L	x	x	x	x	x	x	x	x	x
Column address and read command	READ	H	X	L	H	L	H	V	L	V	L	V	L	V	L	V
Read with auto-precharge	READ	A	H	X	L	H	L	H	V	H	V	H	V	H	V	H
Column address and write command	WRIT	H	X	L	H	L	L	v	L	v	L	v	L	v	L	v
Write with auto-precharge	WRIT	A	H	X	L	H	L	L	V	H	V	H	V	H	V	H
Row address strobe and bank act	ACTV	H	X	L	L	H	H	V	V	V	V	V	V	V	V	V
Precharge select bank	PRE	H	X	L	L	H	L	v	L	x	L	x	L	x	L	x
Precharge all bank	PALL	H	X	L	L	H	L	X	H	X	H	X	H	X	H	X
Refresh	REF/SELF	H	V	L	L	L	H	X	X	X	X	X	X	X	X	X
Mode register set	MRS	H	X	L	L	L	L	L	L	V	L	V	L	V	L	V

Note: H: V<sub>IH</sub> L: V<sub>IL</sub> X: V<sub>IH</sub> or V<sub>IL</sub> V: Valid address input

Figure 6—The CS\*, RAS\*, CAS\* and WE\* pins encode the synchronous DRAM instruction set.

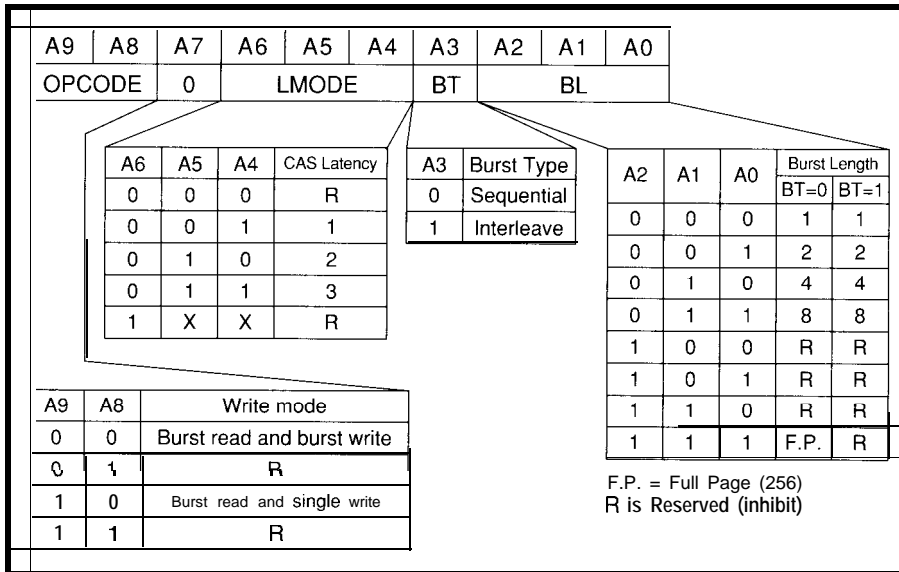


Figure 7-The MRS (Mode Register Set) command defines synchronous DRAM basic operating characteristics such as burst length and CAS\* latency.

which is pretty much like the RAS\* cycle of old. In this case, it selects the bank (A9) and row (A0-A8). This is followed, after the requisite delay ( $t_{RCD}$ ), by a READ or WRIT command that specifies the column address (A0-A7) as in yesteryear's CAS\*. After the previously specified CAS\* latency, the data is blasted on each clock edge.

The READ A, WRIT A, PRE, and PALL commands all have to do with precharge. Since precharge has been automatic since the days of the 2116, it might seem like a step backward to make the system designer handle it.

Fortunately, if you'd rather not be bothered, the A versions of the READ and WRIT commands feature automatic precharge. However, as in plain DRAM, automatic precharge lengthens the basic cycle time. Fortunately, overt control of precharge in conjunction with the bank scheme does offer the opportunity for aggressive designers to hide precharge in one bank while transferring data from another.

That about covers the SDRAM instruction set, except for our old friend refresh, of which two variants are offered, depending on the state of the CKE pin. If CKE is high (i.e., the clock is enabled), an AUTO REFRESH mimics a plain DRAM's CBR-refresh cycle (i.e., an on-chip refresh counter means no external address is required).

SELF REFRESH (CKE low) is kind of a power-down mode, in which the

DRAM keeps itself refreshed while consuming amazingly little power (only 2 mA compared to its 50-100-mA active consumption). It's tempting to just leave the SDRAM in SELF REFRESH between accesses. Unfortunately, however, exit from SELF

REFRESH requires the issuance of 1024 cycles of AUTO REFRESH to update the refresh counter, so SELF REFRESH is only suitable for serious napping.

Figure 9, showing the allowed instruction sequences, looks complicated, but really isn't if you talk it through. When idle, the SDRAM should be refreshed 1024 cycles every 16 ms. An ACTV command (think of it as a RAS\* cycle) selects a bank and row for action. Once activated, the row can be accessed freely, bursting data both ways from random column addresses. After all transfers are completed (or if the row address changes), exit is via a precharge cycle, taking the SDRAM back to the idle and refresh state.

Remember, the key point of the two-bank scheme is that each bank can be in a different state and the state transition intervals are faster between, rather than within, banks. For instance, the minimum ACTV-to-ACTV (i.e., cycle time) spec within a bank is 110 ns while between banks it's only 30 ns.

## TURBO-128

### THE NEXT GENERATION EMBEDDED CONTROLLER

**\*\* NO DEVELOPMENT TOOLS REQUIRED \*\***

READY TO PROGRAM IN BASIC OR ASSEMBLY

**Photronics Research** introduces the T-128: A True Single Board BASIC Development System. The T-128 is based on Dallas Semiconductor's new 8051-compatible DS80C320.

With its 2X clock speed (25MHz) and 3X cycle efficiency, an instruction can execute in 160ns: an 8051 equivalent speed of 62.5MHz!!! Equally impressive is the T-128's high-speed NVRAM interface. Any of the 128K RAM may be programmed directly from a PC file through the console: eliminating EPROMs and associated tools. Program Development has never been faster or more convenient, even with the finest EPROM emulator. The T-128 features PORT 0 bias and EA-select for DS87C520 upgrade.



**BASIC520**

- Modified BASIC-52 Interpreter [BASIC-520]
- Now Fast Enough for New Applications
- Stack BASIC Programs and Autorun
- CALL ASM Routines for Maximum Speed

**I/O**

- Three S-bit Parallel Ports
- Two Full-Duplex RS232 Serial Ports
- Decoded Device/I/O Strobes
- 5w/n Bus Connector

**UPGRADE**

- 0887C520 processor (33MHz)
- Instruction cycle: 121ns
- 8.25 MIPS
- 8051 equivalent: 82.5 MHz
- Internal 16K ROM/1K SRAM

**PROCESSOR**

- Dallas Semiconductor's DS80C320
- 30C% more efficient than the 8051
- Three 16-bit Timer/Counters
- 13 Interrupts (6 Ext. 7 Int)
- A second 16-bit Data Pointer
- 384 Bytes of Internal RAM
- Programmable Watchdog
- Brownout Protection
- Power-Fail Reset/Interrupt
- Power-On Reset
- Fully supported by Franklin C51

**MEMORY**

- Entire 128K Memory Map populated with fast NVRAM [64K DATA + 64K CODE]
- All memory programmed on-board
- Partitionable as CODE/DATA/OVERLaid
- Code Space is Write-Protectable
- Satellite Data Protection

Only 5" x 3-3/4" • 500-hole Proto Area • Console/Power connected by a single 4-conductor telephone wire (very convenient)

**Comes Ready to Run with power adapter/cable assembly. Includes utility diskette with DETAILED TECHNICAL MANUAL \$199 in qty.**

109 Camille St. • Amite, LA 70422 • (504) 748-9911 • Tech Support (504) 748-7090 • FAX (504) 748-4242

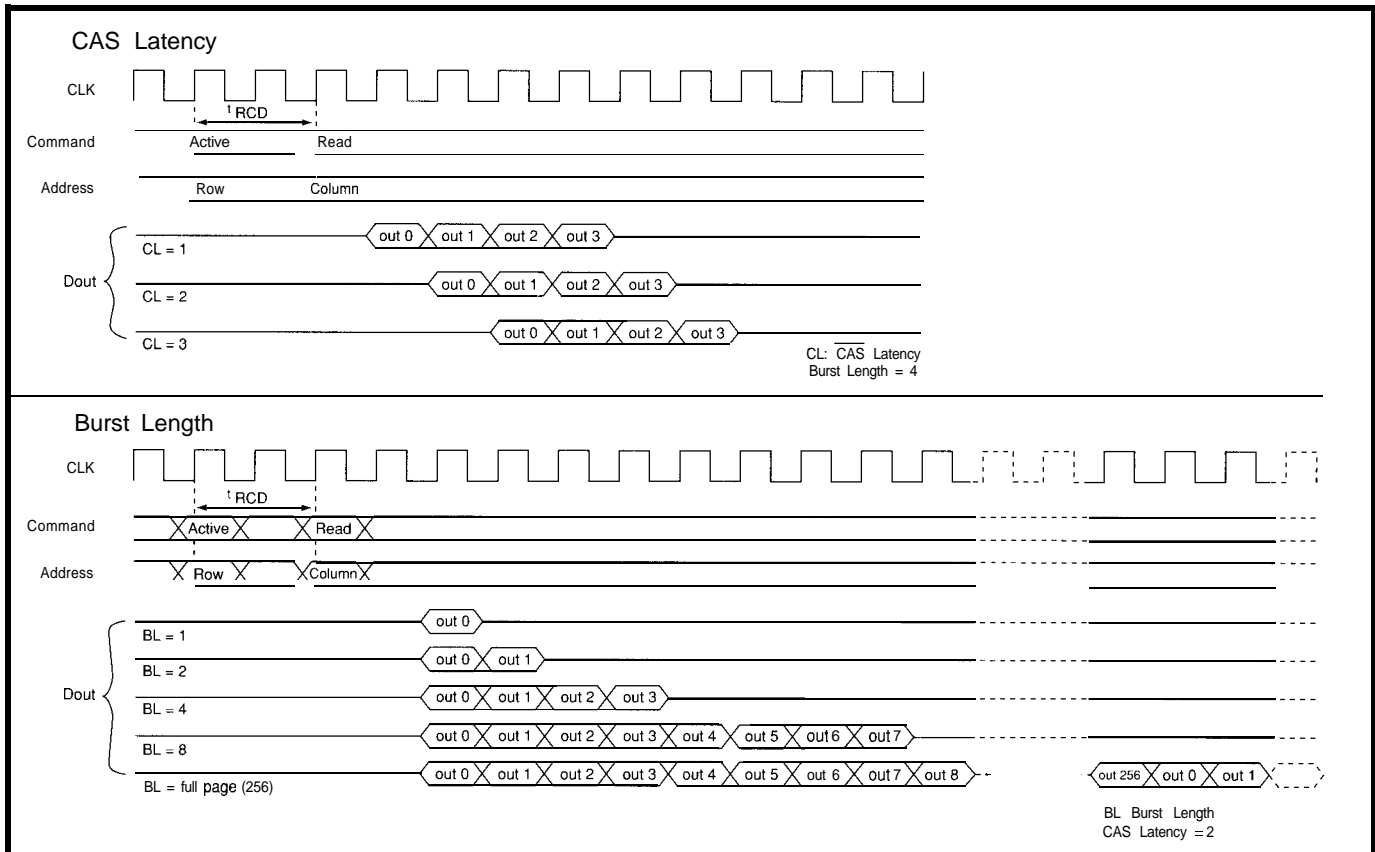


Figure 8—Though not shown on the figure, remember that higher CAS\* latency offers a faster clock rate. Short-burst transfers automatically terminate, but full-page bursts must be explicitly terminated with the BS J command.

**CIARCIA  
DESIGN  
WORKS**

Does your Big-Company marketing department come up with more ideas than the engineering department can cope with? Are you a small company that can't afford a full-time engineering staff for once-in-a-while designs?

Steve Ciarcia and the Ciarcia Design Works staff may have the solution. We have a team of accomplished programmers and engineers ready to design products or solve tricky engineering problems. Whether you need an on-line solution for a unique problem, a product for a startup venture, or just experienced consulting, the Ciarcia Design Works is ready to work with you. Just fax me your problem and we'll be in touch.

REMEMBER... A CIARCIA DESIGN WORKS!  
Fax (203) 871-8986

**Unleash  
the  
Power  
of  
PromICE**

- ❖ Connect via ROM Socket; DIP; PLCC; SMT
- ❖ Emulate ROMs up to 16MBit in Size
- ❖ Fastest Downloads Available:  
Parallel; Serial; Ethernet
- ❖ Run Industry Standard Debuggers
- ❖ Target Processor Independent
- ❖ Support 3Volt Targets
- ❖ Host Software Sources Included
- ❖ Shielded Cables for Reliable Operation
- ❖ 30-day Money-Back Guarantee;  
1 Year Warranty!
- ❖ Unlimited Phone Support; 24hr BBS

**Call Today 1-800-PROMICE**  
(1 ■ 800=776=6423)

**Grammar Engine Inc.**  
921 Eastwind Dr., Suite 122 • Westerville, OH 43081  
614/899-7878 • Fax 614/899-7888

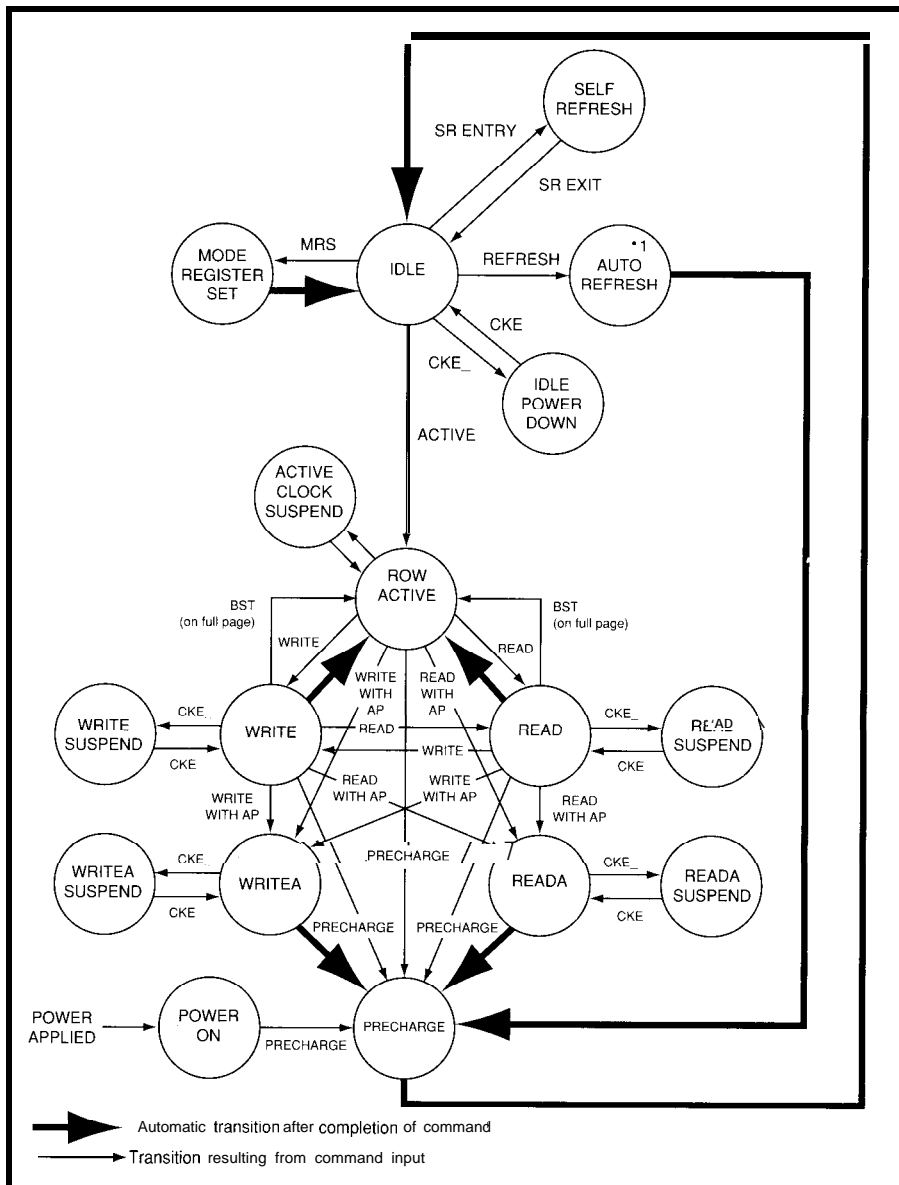


Figure 9—An synchronous DRAM is either idle or actively reading and writing or precharging.

## BURST OR BUST

You might think all this 66- and 100-MHz talk implies some kind of lo-times breakthrough in DRAM cell speed. In fact, the speed of basic memory operation really isn't much different (witness the 110-ns cycle time mentioned above is about the same as plain DRAMs). The SDRAM simply blazes a faster trail to what's otherwise regular DRAM.

Thanks to the ping-pong feature, even the simplest SDRAM setup (CAS\* latency = 1, burst length = 1) offers good potential. Note that the parallelism can be extended across multiple SDRAMs as well (i.e., you can also ping pong between chips).

However, there's no doubt that bursty applications such as high-end, 32-bit CPUs and graphics best match the features of SDRAM. The tradeoff between CAS\* latency, clock rate, burst length, number of chips, and so on is hard to generalize.


Graphics applications will likely shoot for maximum bandwidth (e.g., CAS\* length = 3, burst length = full page, CLK = 66 MHz). It then becomes simply a matter of choosing the number of SDRAMs to support a given display. For example, a 1024 x 768 x 24 x 72-Hz display calls for 170 MB per second, while each 66-MHz SDRAM offers 132 MB per second. Thus, a two-SDRAM frame buffer easily handles

display refresh, leaving about 40% bandwidth for drawing.

CPU main-memory applications are a little more tricky. A simple controller may just set the burst length equal to the cache-block-refill size and leave it at that. CAS\* latency versus CLK rate largely depends on memory-controller speed, cost and complexity issues, and "hit ratio" tradeoffs (i.e., weighing greater latency penalty against higher burst bandwidth).

More aggressive controller designs are possible if you're willing to sift through the entrails of your program and truly understand its behavior. It might be wise to put instructions and data in separate SDRAMs to take advantage of locality. Often, instruction- and data-cache-refill sizes are different, so each SDRAM's burst length could be set appropriately.

Another idea is to control burst length dynamically, either by overtly terminating full-page bursts or by reprogramming the mode register. This opens the door to really clever dynamic or "greedy" burst schemes, which wring all possible bandwidth out of the SDRAM.

No doubt, the prospect of rolling your own SDRAM controller, especially a fancy one, is rather daunting. The good news is I expect SDRAM support will quickly migrate into likely partners including 32-bit RISC CPUs, graphics ICs, and PC chipsets. 

*Tom Cantrell has been an engineer in Silicon Valley for more than ten years working on chip, board, and systems design and marketing. He can be reached at (510) 657-0264 or by fax at (510) 657-5441.*

## CONTACT

Hitachi America, Ltd.  
Semiconductor and IC Division  
2000 Sierra Point Parkway  
Brisbane, CA 94005 18 19  
(415) 589-8300  
Fax: (415) 583-4207

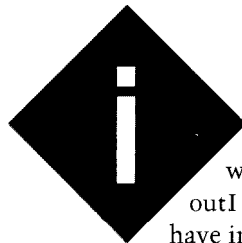
## IRS

422 Very Useful  
423 Moderately Useful  
424 Not Useful

# Downsizing

## EMBEDDED TECHNIQUES

John Dybowski



It's funny the way things work out. I would never have imagined that

someday I'd be a proponent of the 8051 architecture. But, in light of the work I've been doing lately with a variety of 8051 derivatives, this must be exactly the way I'm coming across. Quite frankly, I'm not entirely comfortable with the situation.

I still remember my first encounter with the 8051. I immediately recognized its utility as a Boolean processor and was quite impressed with the ease with which it handled bit-oriented I/O. At the same time, I was appalled by the seemingly random architecture and the absence of some rather fundamental instructions. These idiosyncrasies continue to irk me and appear all the more enigmatic in light of recent microcontroller advances.

Still, it pays to remember that silicon was a much more precious commodity at the time the 8051 was developed than it is today. Obviously, the seemingly arbitrary exceptions and bizarre architectural quirks were the result of gut-wrenching decisions made to bring the processor's principal features in line with a constrained silicon budget. It's not an uncommon problem. I'm sure the consequences of those unpleasant decisions were not taken lightly.

Although the mechanics of crafting silicon have changed, the fundamental aspects of doing business remain the same. And, this is the key to the puzzle since this is, after all, a business venture and not a science fair project. As a case in point, consider

how this same scenario has replayed over the years.

More recently, the same tough choice between price, performance, and features was played out with yet another strange, but extremely successful microcontroller—the much maligned, Microchip PIC 16C54. To me, this processor is the epitome of business sense winning out over technological constraints.

The thing doesn't even have an interrupt. Although technically a no-brainer, adding an interrupt would have resulted in totally blowing the target price. Talk about tough choices. As it turned out, the chip's many deficiencies were each countered by a combination of skillful marketing and imaginative technical support. They even convinced a sufficiently large number of engineers that they really didn't need that interrupt after all!

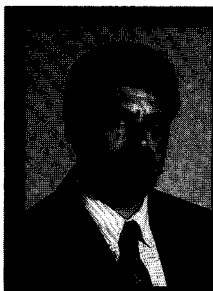
Both the 8051 and the 16C54 architectures have established themselves in their respective spheres of application. As Microchip develops the PIC series into larger, more capable devices, it's inevitable that the architecture's basic simplicity has become somewhat obscure.

Ironically, the 16C54's big brothers are starting to look a lot like 8031s, a rather questionable distinction. It's also a dubious arena to compete in since it is well-served by some firmly established controllers. To me, the original 16C54 is still the soul of the PIC family. Warts and all, this is the part I turn to when I need to implement bare-bones functionality and all I can afford is a \$2 chip.

### WRAP IT UP

Formerly, using a higher-level language on 8051-class processors was considered a disputable practice at best. After all, the basic architecture has little enough real stack space to begin with and no stack manipulation capability at all—obviously, a major drawback with a stack-oriented language. Combining these handicaps with a meager instruction set and all those overlapping code, data, and bit segments, it's no wonder that many viewed using code generators on such micros as nothing more than an

## Atmel's AT89C205 1 Flash-based Microcontroller



This little chip is essentially a flash-

based 8051 with some extra RAM in a 20-pin package. True, it does possess fewer I/O pins, but it provides all the peripherals and functions of a standard 8051.

interesting diversion with little practical consequence.

Evidently, in spite of these obstacles, compiler developers have managed to prevail. Actually, it could be argued that it is because of these obstacles that they have been so successful. The fact is, placing a good code generator between such an architecture and yourself isn't that bad an idea. When I'm writing code, I don't mind dropping to the machine level when it's necessary or expeditious. I'm also well aware of how counterproductive it is to remain at that level any longer than necessary.

An analyses of the situation reveals that some of the most popular embedded-C compiler implementations run on some of the more "difficult" processors. This could be construed as something of a paradox. The implication seems to be that these popular, although irregular, processors assume a level of anonymity through the insulating effect of the language compilers.

From this, it follows that selecting an appropriate architecture should be a relatively straightforward and rational exercise. This might be true if a typical embedded program could be coded entirely using a higher-level language. That this is not the case, especially with 8-bit processors, is evident by the continued popularity of older architectures like the ones I've been describing—so much for rational thinking. If it looks to you like a case of the tail wagging the dog, then you're not alone.

The fact is, as any embedded developer well understands, portable code is something that is extremely elusive and difficult to accomplish in most embedded applications. Usually, system I/O turns out to be such a big part of the overall picture that it is the primary concern.

And, regardless of "proper" coding techniques, embedded programs are often laced with timing dependencies that rely on the execution times of various pieces of code. When doing this type of programming, the bottom line is that you have to know when to get down to the machine level and you must have a good understanding of the

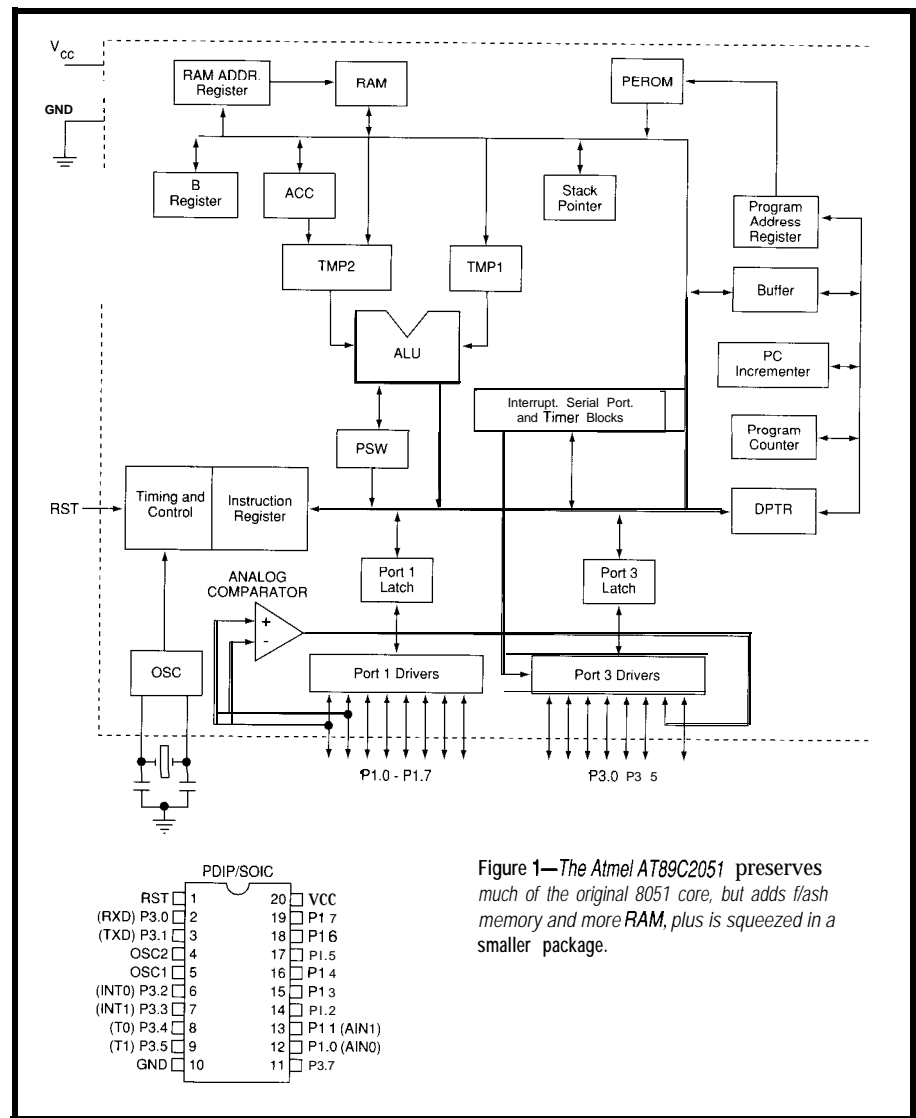


Figure 1—The Atmel AT89C2051 preserves much of the original 8051 core, but adds flash memory and more RAM, plus is squeezed in a smaller package.

quality of code your compiler is generating.

Still, in spite of all the complicating issues, the productivity gains that can be realized from using a higher-level language offer a compelling incentive to rethink the way you've been writing your embedded programs.

The benefits of using language compilers can be perceived differently depending on your particular needs. Minimally, an efficient compiler can function as a sort of stylized assembler. This is especially true of the C programming language which affords excellent control of low-level functions. The incentive for this category of usage is pretty strong with many of the more limited controllers and processors.

Recently, some of the most difficult C implementations have been

realized on resource-starved controllers such as the PIC16C54. Frankly, when I first found out that engineers were actually seeking C compilers for the 16C54, I was a bit baffled. I mean, with a two-level stack, 512 words of program memory, and a few dozen bytes of read/write storage, it didn't exactly seem like a good fit. It looked like the compiler's overhead alone could swamp the processor's meager resources.

It turns out that there are some very smart software people and now there are a number of C compilers available for these minuscule processors. Reports from the field indicate that this stuff really works. I haven't tried one yet but, having done a couple of projects involving the 16C54, I'm very interested. You see, one of the big attractions of a language compiler for

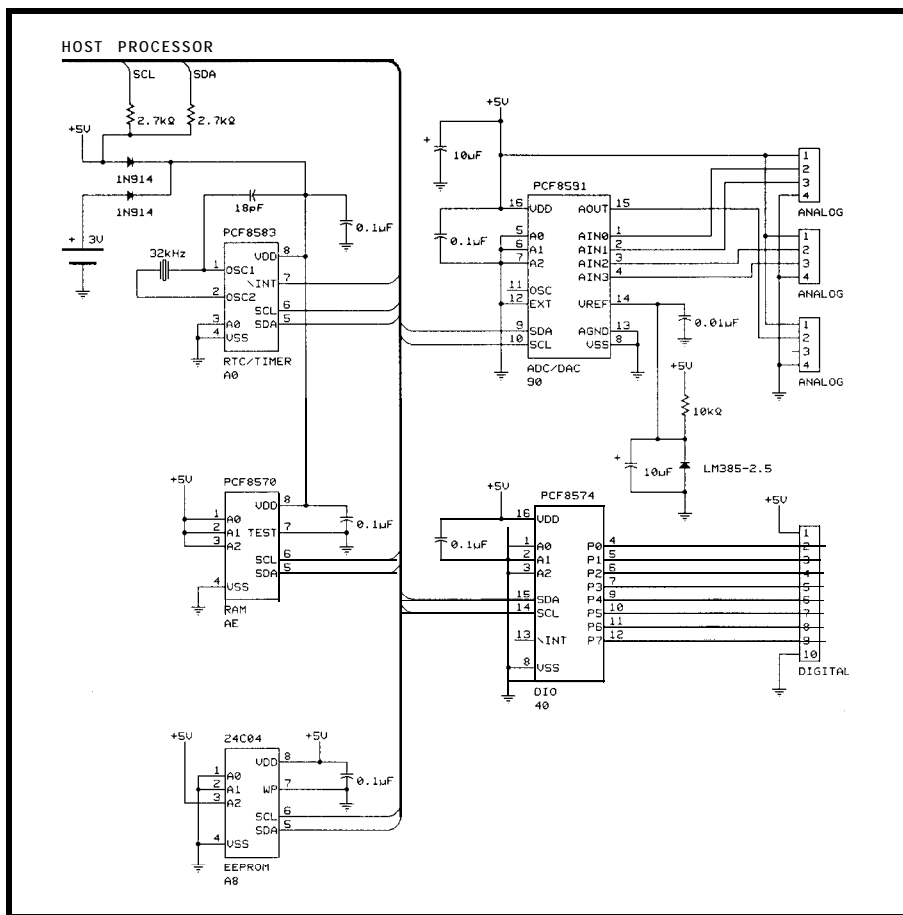


Figure 1—Host I/O bits are preserved by connecting most I/O devices to a single pair of PC lines. The I/O board includes a real-time clock and timer, RAM, EEPROM, analog in, analog out, and digital I/O, all connected to just two wires.

me is its ability to handle many of the tedious programming details. And, there are many of these indeed when working with very small processors.

Put another way—if it's ugly, wrap it up!

## LISTEN TO YOUR CUSTOMERS

Recent evolutionary trends have involved enhancing "standard" processor architectures. For better or worse, the 8051 is now perceived as standard indeed. In recent columns, I've shown how this basic architecture has gained in performance and capability while retaining its standard footprint and pinning.

There's also been a trend to downsize the architecture. Downsizing has led to a loss of some capabilities, a reorganization of the fundamental processing core, and a reduction of on-chip memory. The high end has been well served by a variety of capable derivatives. The low end until recently has been... well, the low end. But

finally, someone has introduced a very small 8051 that doesn't give up any of its fundamental features.

The new, flash-based AT89C2051 from Atmel is a welcome addition to the 8051 family. More than just another scaled-down 8051, this processor is a full-blown 8051 housed in a 20-pin package. All the standard 8051 peripherals have been retained including two 16-bit timers along with their gating controls and associated interrupts, two external interrupts, and a full-duplex hardware UART. This is in addition to 2 KB of flash program memory and the standard 128 bytes of internal RAM.

It may not seem like a big deal at first, but it is the inclusion of the full 128 bytes of internal RAM that opens up substantial opportunities for using this part with standard commercial C compilers. While using a compiled language is certainly an option with some of the more restricted 8051 derivatives, things naturally get quite

limited with the 64 bytes of RAM most of these smaller chips offer. And, if you're worried about code efficiency, the AT89C2051 specifies an operating frequency from DC to 24 MHz. This gives you the elbow room to handle any reduction of efficiency that results from using a code compiler.

Another area that competitive products have consistently ignored is the inclusion of a hardware UART. Although many of the competing parts offer hardware-assisted I<sup>2</sup>C interfaces, it's obviously of little consolation when you have to provide for standard asynchronous communications.

From where I sit, the outcry for a hardware UART has been long, loud, and clear. Needless to say, thus far, the semiconductor manufacturers have been totally unreceptive to what users have been calling for. Their response to these requests has simply been to either say that you really don't need it or to suggest running one entirely in firmware. I'm surprised this "what we've got is what you need" mentality has prevailed as long as it has.

As a result of this attitude, manufacturers have left themselves wide open, jeopardizing a significant market share. If the AT89C2051 offered nothing more than just a hardware UART, its success would be ensured. The fact is it's got a lot more than that going for it. Figure 1 shows the contents of this 20-pin processor in block form.

## I LIKE I<sup>2</sup>C

The AT89C2051 possesses a full 8051 architecture, it just happens to have fewer I/O pins. Obviously, to design a general-purpose embedded computer around such a device, it's necessary to conserve these I/Os. The most efficient way to hook up a lot of peripherals using just a couple of lines is over the I<sup>2</sup>C bus.

At first, it might seem that not having any hardware I<sup>2</sup>C support might prove to be a major liability in what essentially amounts to an I<sup>2</sup>C-based computer. But, with the exception of some very specialized applications, it is not a problem at all.

Remember that I<sup>2</sup>C defines a synchronous protocol that is capable of

running all the way down to DC. With the AT89C2051's complement of internal and external interrupts, it's easy to arrange time-critical processes to run entirely under interrupts. This lets you bit-bang the I<sup>2</sup>C and still stay live to service your real-time operations.

Obviously, this is luxury you don't have when bit-widdling asynchronous communications, which by definition must adhere to relatively strict timing constraints. Sure, you can run asynchronous communications in the interrupts, but there are serious limitations to the maximum bit rate you can reach.

Some I<sup>2</sup>C peripherals come prepackaged and ready to go. The small multifunction card shown in Photo 1 illustrates a number of such useful functions. Figure 2 depicts the card's schematic. The I/O-related functions include four single-ended or two differential 8-bit analog inputs, one 8-bit analog output, and eight digital I/O points.

The card also provides a real-time clock-calendar timer with 256 bytes of RAM, a dedicated 256 bytes of RAM, and 5 12 bytes of E<sup>2</sup>PROM. The RTC and RAM are backed using a BR1225 lithium coin cell that is also contained on the card. The card, including connectors, measures only 1" x 3". If your application doesn't need these functions, then simply don't load the card. PC, especially a firmware-based implementation as used here, carries very little hardware overhead. Even the requisite pull-up resistors are located on the card.

Higher-level I<sup>2</sup>C functions are realized using a combination of I<sup>2</sup>C and logic ICs. Using a couple of I<sup>2</sup>C-to-parallel converters

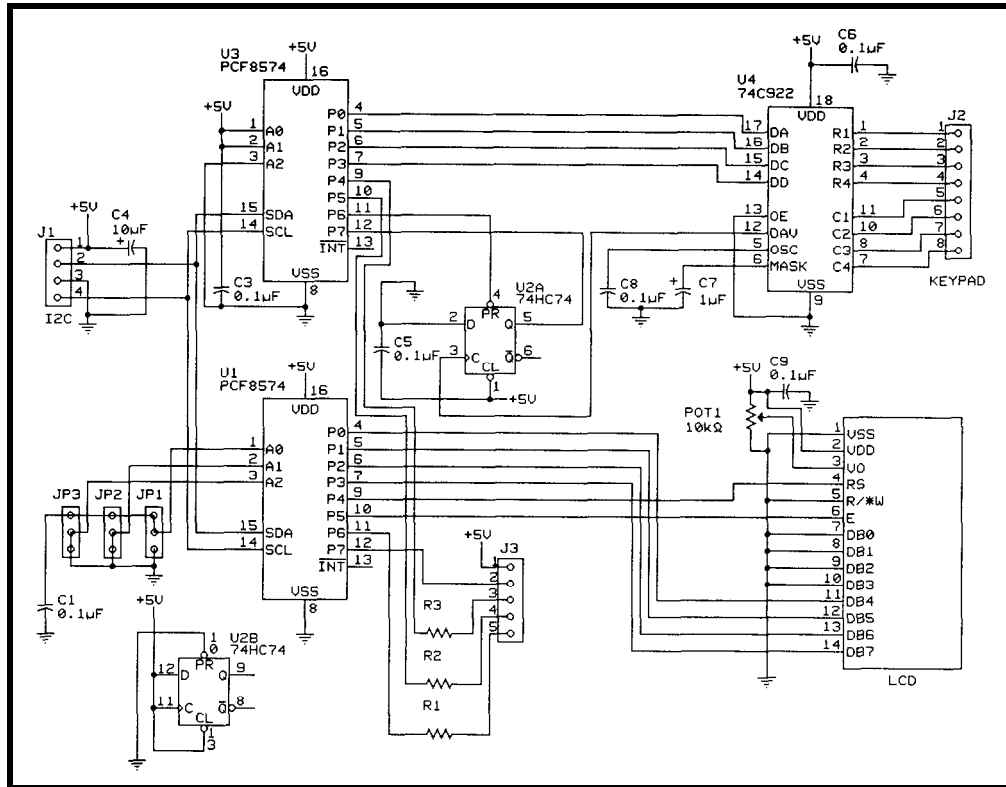


Figure 2b—Another interface board handles a 4 x 20 LCD, 4 x 4 keypad, and general I/O over two wires.

and some support logic, a 4 x 20 LCD and a 4 x 4 keypad can be supported using the same two I/O lines. The control card for this peripheral system attaches almost invisibly beneath the standard 4 x 20 supertwist LCD panel as shown in Photo 2. This arrangement leaves a number of general-purpose I/O lines free to drive a beeper, LEDs, and other indicators.

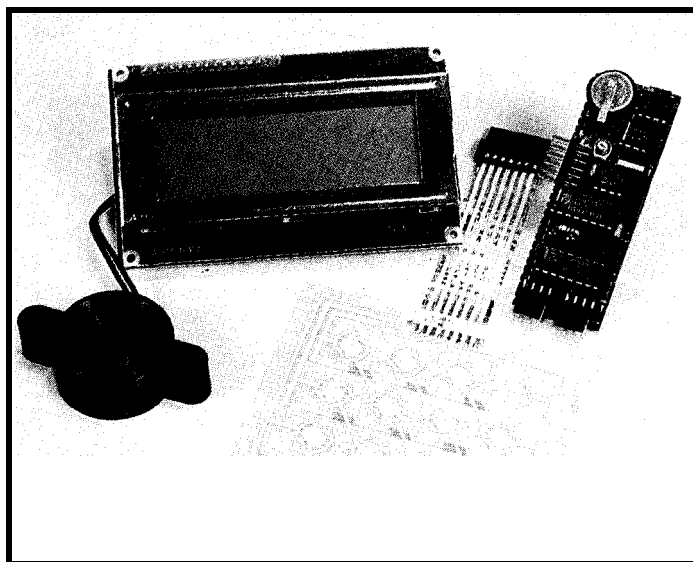


Photo 1—A sample of the peripherals which can be easily hooked up using a combination of PC and logic ICs. All the devices shown in Figure 2a fit on a 1" x 3" board (upper right) while those in Figure 2b fit under the LCD display.

## PROCESSOR I/O

So far, the I/O devices I've described encompass all the peripheral functions a very small embedded computer might need. The beauty of this arrangement is that, although the 89C2051 has only 15 I/O pins to begin with, after accommodating all these functions we are still left with 13 for general usage. These I/Os can be

directly tested and set by the processor and are therefore very fast. Among these general-purpose I/O bits are two external interrupts, receive-and-transmit connections to a full-duplex UART, and the gating signals for the two internal 16-bit timers.

Although the on-chip I/O lines are for the most part 8051 compatible, there are some important differences. Refer again to Figure 1. You can see the familiar 8051 pinout nomenclature has been preserved. From this figure, you can also see



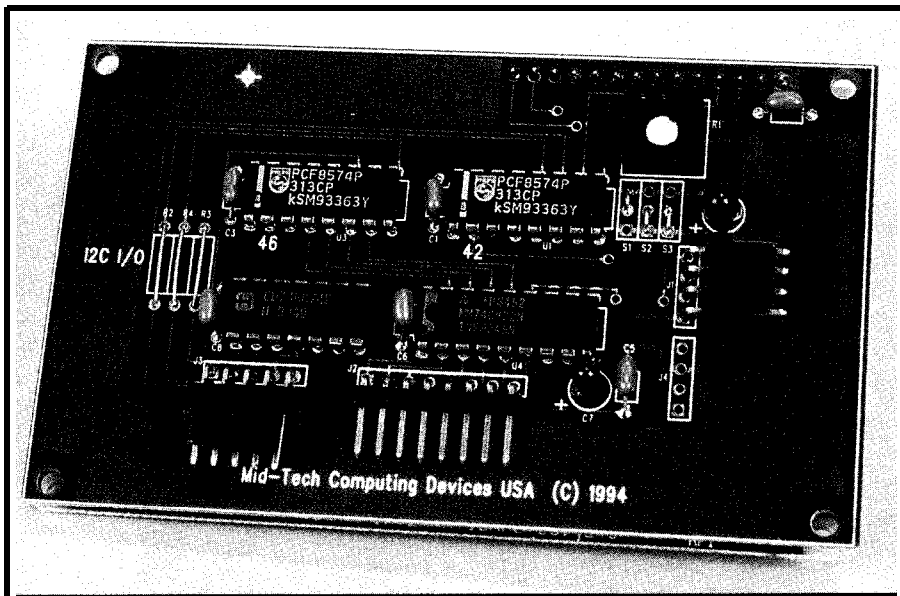


photo 2-A standard 4 × 20 LCD display, 4 × 4 keypad, and PC interface are supported on a board that mounts on the back of the display.

that pins P1.0 and P1.1 are designated as analog inputs. These high-impedance inputs serve as positive (AIN0) and negative (AIN1) inputs to the built-in precision analog comparator. P3.6 is not available externally and instead takes the output of this comparator as its input. To facilitate interfacing to real-world loads without the need for external buffers, ports 1 and 3 are rated to sink a full 20 mA when running at 5 V.

## MAKE SENSE?

If you accept that the peripheral mix I've described for my system is not unusual, then perhaps I can offer some insight from my own experiences on issues like code portability, code reuse, and the enduring success enjoyed by the 8051 architecture. In this context, I will also elaborate on why I feel the 89C2051 is such a significant processor.

Simply put, the I<sup>2</sup>C peripheral set I've described has served me well in a variety of configurations on a variety of host systems for a number of years. I've developed a fair amount of C code to support these functions, but the main low-level drivers are all, of necessity, written in hand-tweaked assembler.

This in itself represents a significant investment of time and effort. To this, of course, you must add the

“standard” overhead of the trade. This overhead includes such things as interfacing low-level code to the compiled code (usually quite different for each compiler you use); debugging, testing, and validating functions; documentation; and continuing support. Underestimating these associated tasks can have a deleterious effect on your continued success in this field!

Electronics means different things to different people. To those of us that have bills to pay, it's a business. In this context, the bottom line revolves around spinning off applications. Efficiency in doing so is what can make or break it. Sure I could code up a new set of low-level drivers for a different processor in a relatively short period (disregarding the overhead of course). And, if I had to, I could take all that C code and rewrite it in assembler if the processor I was using didn't have the horsepower to handle a compiled language.

Individually, these are fairly small tasks. It's only when you look at the overall picture that it becomes evident that a significant amount of time can be spent on these tasks. And, I haven't even touched on things like regenerating all my interrupt-driven communication routines, timer-capture functions, and the like. There comes a point when you have to ask yourself

where you want (or can afford) to spend your development time.

## EMBEDDED TOOLS

I'm sure it's obvious I have more than just a passing interest in the new Atmel processor. Next month, I'll cut through the fluff and devote my attention to the design and development of a very small, general-purpose computer based on the AT89C2051.

Having defined the 1" × 3" form factor for the I<sup>2</sup>C peripheral card, I will use the same footprint for the main processor card. This card will contain the 20-pin, flash-based processor; an RS-232 and RS-485 interface; power supply; and prototype area.

The main focus of my column, however, will be the associated development system for the AT89C2051. In this respect, I'll also be covering the AT89C51 flash-based processor that, with some hardware assistance, emulates the AT89C2051. The system works with a variety of target systems since it uses a standard umbilical cable hookup.

Of course, once you get an application up and running, you'll want to program the AT89C2051 flash memory. This will be accomplished with the built-in flash programmer contained right on the development system. As usual, I'll be calling on my esteemed colleague, Dave Dunfield, for his fine PC-hosted development tools. I'll be using his simulator and remote-debugging software to round out the development system. For code generation, I plan to use his Micro-C and assembler products. ☐

*John Dyhowski is an engineer involved in the design and manufacture of embedded controllers and communications equipment with a special focus on portable and battery-operated instruments. He is also owner of Mid-Tech Computing Devices. John may be reached at (203) 684-2442 or at john.dyhowski@circellar.com.*

## IRS

- 425 Very Useful
- 426 Moderately Useful
- 427 Not Useful

# CONNECTIME

conducted by Ken Davidson

The Circuit Cellar BBS  
300/1200/2400/9600/14.4k bps  
24 hours/7 days a week  
(203) 871-1988—Four incoming lines  
Internet E-mail: [sysop@circellar.com](mailto:sysop@circellar.com)

*We're going to stray a bit from hard-core electronics this month to enter the more physical world. In the first thread, we look at cable impedances, connectors, and termination. The topic can be something of a black art at higher frequencies.*

*In our only other thread, we look at some suggestions for running a three-phase motor on single-phase power. The ideas run the full gamut of implementation complexity, and is a fun discussion to read through.*

*Keep an eye out next month for the next version of TBBS (2.3). It promises to overhaul the file area support, plus adds some nice, small enhancements throughout the rest of the system. I'll tell you more once we have it up and running.*

## Cable termination

**Msg#: 5925**

**From: DAN HOPPING To: ALL USERS**

Not being the video analogical type, I have a couple of questions on the subject of video impedance and termination. I have a fancy video edit control system that brings the video in from two player decks and outputs the mixed signals and effects to the record deck. I have discovered that there are short (less than 12") 50-ohm coax cables that bring the video signals into the PC and back out.

My questions are: Would this short length of 50-ohm cable make a difference to the video signal that, as I understand it, should be terminated with 75 ohms?

What does it mean to have a 75-ohm cable versus a 50-ohm cable? Are there different BNC connectors for 75-ohm versus 50-ohm cables? What does impedance mismatching actually do to the signal?

**Msg#: 5967**

**From: LEE STOLLER To: DAN HOPPING**

A 75-ohm cable is standard for video signals. The BNC connector you use is determined by the actual cable type. For runs that are not very long, cheap RG-59U cable works out well enough, and there are BNCs designed to mate with it. For long runs in professional installations, type 724 cable is often used. This is much thicker, and requires BNCs designed for it. The cables going to the editor will probably be terminated inside the editor. If there are two BNC jacks for each cable, and there is no termination switch on the

editor, connect the cable to one jack, and then connect a "75-ohm BNC termination" to the other. There is a similar situation at the input to the record machine.

If, as you say, a cable takes video both to and from the PC, the PC connection is a "loop-through," and the necessary termination would be located at the other end of the "from" cable.

**Msg#: 5933**

**From: RUSS REISS To: DAN HOPPING**

The impedance of a cable (any cable, coax or otherwise) represents its properties in propagating a wave down it. Unlike DC signals, you must take this into account when sending high-frequency signals down a cable.

Consider a line terminated in a variable resistance at its far end. First take the two extremes of an open circuit and a short circuit. With an open, there can be no current flowing at the far end (no path for it to flow), but voltage can be present. With a short, there can be no voltage developed at the end, but current can flow. Impedance is the ratio of voltage to current anywhere on the line.

If a voltage (or current) wave is propagating down a line and sees the open or short, interesting things will happen. In order to meet this "boundary condition," a reflected wave of suitable voltage/current magnitude will be created. This wave will propagate back down the line from the far end to the sending end, and will vectorially add to the wave going in the other direction.

Recall that this propagation takes place at nearly the speed of light, just a bit less depending on the "propagation velocity" of the cable (often in the 60-80% range). This too depends upon the cable's physical properties such as conductor size, spacing, and dielectric constant of the insulation. The fact that the wave does not propagate *instantaneously* is why all these considerations are necessary. For DC or very low frequencies, the propagation time (on most length cables) is very short compared to a wavelength of the signal. That's not true at higher frequencies.

Returning to the open/short condition, what effect that sort of termination has at the *sending* end (which is often where you are most concerned) depends on the electrical length (in wavelengths) of the line. For example, an "open" placed any odd quarter wavelengths away from the sending end will actually look like a short, while a "short" will look

# CONNECTIME

like an open! Interesting, huh? At multiples of a half wavelength, the load will reflect back looking exactly like what it truly is. At in-between wavelengths it can look capacitive or inductive. Life gets interesting!

When a wave propagating down a line sees a load equal to its characteristic impedance, the voltage/current (hence impedance) relationship all along the line is satisfied when the wave reaches the far end. Thus the signal "sees" a line that (electrically) appears to go on forever and ever! This is what's referred to as "matching impedances" and why a 75-ohm video source usually is sent over a cable with 75-ohm characteristic impedance and is terminated in a 75-ohm load. Ditto for 50-ohm source, 300-ohm TV twin-lead, or even 120-ohm twisted pair used in, say, RS-485.

What effect will a short cable of different impedance have? It depends on the frequency (wavelength) of the signal. For your video example, using a short 50-ohm cable of a few feet will usually have no effect. Same goes for connectors.

The physical "length" of a connector is quite small, so using the "wrong impedance connector" only becomes a problem when that length is a substantial portion of a wavelength. In practice, this means above 100 MHz or so. Any BNC connector will suffice for 50/75-ohm systems below that frequency. But there *are* both 50- and 75-ohm BNCs (with slightly different physical dimensions, usually of the center pin area) for precise matching. You don't have to worry about it in your case.

Often, video equipment (such as monitors, etc.) include a switch which permits the 75-ohm terminating resistor to be switched in or out. If you multidrop more than one device on the cable, you want only the last or end device to have the terminating resistor present (per the above discussion), while the others operate in "high-impedance" mode and have little effect on the signal as it passes by. Again, same is true with RS-485 multidrop communication.

**Msg#: 6037**

**From: DAN HOPPING To: RUSS REISS**

Thanks for your wonderfully detailed reply. Your explanation helped a lot! I usually get a little lost before I see the entire picture because I really like to understand. Let me put this in terms a digital guy can understand and you tell me if I've missed it. I understand the process of terminating an address line on a digital board to reduce reflections and ground bounce. The same holds true for local area networks and RS-485 communications. So, does the following make sense?

The whole deal with impedance matching is to provide a signal path that will not cause reflected waves from an earlier signal to either attenuate or reinforce the actual signal as it reaches a device in the signal path.

Assuming the previous to be true. If you had a video transmitter at the start of the cable and a video monitor (receiver) at the end of the cable, why couldn't you just use any shielded cable and terminate the line at the receiver with a 75-ohm resistor? Would the terminating resistor not eliminate to ground any reflectable portion of the remaining signal or must the entire length of the cable (assuming its length is significant compared to the wavelength of the signal) have the characteristic impedance to properly eliminate the reflection? Finally, is there anything more than reflected interference to be considered (assuming the cable can handle the power)?

**Msg#: 6057**

**From: RUSS REISS To: DAN HOPPING**

Dan, you've pretty much got it. Yes, it's precisely the same as terminating address lines or RS-485 buses. By the way, the same holds true for any wave propagation situation, not only electrical waves-acoustic, thermal, optical, and so forth. But keep in mind that the medium (cable or air or water or whatever) always has a characteristic impedance of its own. If you match the transmitter and receiver to this characteristic impedance, then there are no reflections in the system.

So, your example of a video source (with 75-ohm output impedance) connected to a video receiver (high impedance) swamped by a 75-ohm terminating resistor is quite correct. But, the cable that connects the two should also have 75-ohm characteristic impedance for everything to match. If the run is very short (compared to a wavelength) then there are still reflections, but they occur in a very brief time, usually much less than any frequencies of interest, and are essentially filtered out.

I might point out, this "wave mechanism" exists for DC signals too. If you send a step function from a signal generator (battery and "clean" switch) with a sending impedance equal to the characteristic impedance of the line, you can observe some interesting effects on a fast scope as you vary the terminating impedance. The voltage will propagate down the line, hit the mismatch, cause a reflection, come back and vectorially add to the outgoing wave, and repeat this over and over, creating interesting patterns. Often this is taken to be the "ringing" people see on digital signals (e.g., address lines) and is why terminations and even "transmission line" circuit traces are used with fast ECL logic, for example.

Another interesting property of impedance matching, while we're on the subject, is that two lines of different characteristic impedances can be "matched" (connected properly together) by a quarter-wave line with a characteristic impedance which is the geometric mean (square root of the product of the Z of the two lines). This capability is

# CONNECTIME

used in many optical systems as well as in sonar systems, say for matching a transducer to the characteristic impedance of air or water. Just a matter of finding a material (transmission medium) of the proper  $ZO = \sqrt{Z1 * Z2}$  and making it the right "length" for a given frequency. It only works at one frequency (and odd multiples of that freq), but quite useful. Welcome to the fun world of RF.

**Msg#: 6076**

**From: DAN HOPPING To: RUSS REISS**

Thanks again, Russ. I understand the idea but I still have one puzzler floating around in my mind. I understand the idea of the medium damping the propagation of a wave at a given frequency (air, water, whatever...). What still puzzles me is your phrase "isn't the reflected wave of the same frequency" and if so, why does the cable then tend to attenuate the reflected signal?

In all of the RS-485 designs I have done I use simple 24-gauge solid telephone T-wire. The characteristic impedance of that stuff has to be all over the place. It gets twisted, looped, and knotted by installers all the time, yet if I terminate the final drop correctly I don't have any problems no matter how long the run. Am I just attenuating the reflection enough to make everything work and still actually have reflected noise on the line?

I have never looked at the signal with a fast scope to see this (as I say, no problems, so I didn't know I needed to look). Would a cable with a specific characteristic impedance (whatever it is for my data rate) be better than the wire that is recommended? Maybe it's just a cost thing, and my wire works well enough.

**Msg#: 6084**

**From: RUSS REISS To: DAN HOPPING**

Dan, "attenuation" is something quite aside from all I've discussed. Reflections take place due to impedance mismatch. Attenuation comes from lossiness (resistance or radiation) of the medium. You can (and DO) have reflections, even from a lossless line (or other medium) if there is a mismatch.

Yes, T-cable is a somewhat inspecific ZO. It's normally taken to be around 100-150 ohms, and I often use 120-ohm terminations in things like RS-485. You're correct, a small amount of mismatch does little harm, which is why it's always worked for you, and why you've never felt the urgency to investigate further. Also, when the line is short compared to a wavelength (which, so often, it is), the effect isn't very serious-though present.

What you are doing is correct and works. Don't get overly concerned from all our discussion here. It just gives you some background to further understanding. Remember, this all started with you being concerned over use of lines of

different characteristic impedance. No knowledge often leads people to do things incorrectly; a little knowledge can sometimes be worse because it leads to unwarranted fears; full knowledge (is there such a thing?) can put your mind to rest that all is well-and will stay that way!

---

## Three-phase motor on single-phase power

**Msg#:36086**

**From: PETE CHOMAK To: ALL USERS**

Does anyone know of an easy and cheap way to run a 1-HP, 3-phase motor from single-phase power? No rotary phase converters or fancy inverters, because it would be cheaper to replace the motor considering its small size.

**Msg#:37103**

**From: CALVIN KRUSEN To: PETE CHOMAK**

From my experience, a 3-phase motor will run on single phase. The problem is it won't start off of single phase. The cheapest thing to do would be to apply power to the motor, then start it spinning in the direction you want. I have done this in a bind. Also, I don't know how much you would have to derate the motor running on only single phase.

For liability reasons, I suggest you do this as a test only!

Other than the above, or using an inverter, I don't know what to do.

**Msg#:38447**

**From: PETE CHOMAK To: CALVIN KRUSEN**

A while back someone said something about using some kind of capacitor setup to "shift" a phase for small motors, but that's all I remember. Plus, because it is only a 1-HP motor, whatever I do has to be simple to be worth it.

As far as derating, it probably is not critical. The motor is on a Bridgeport I just got, and I do have a three-phase generator to use for power if I need to before I am able to get some sort of power or new motor for permanent use. The motor is a "Fairbanks-Morse" induction motor, 3-phase, 60-Hz, 220/440 V unit if that gives anyone any ideas.

**Msg#:39862**

**From: MIKE TRIPOLI To: PETE CHOMAK**

I have a friend that ran his machine shop out of his garage when he started up. He had the same problem you have in running his Bridgeport. You'll need 220 VAC available. Here's how we addressed it. It's going to sound a little odd, but it works great. Here goes..

Get a washing machine motor and put a pulley on the shaft. Mount it to a bracket and a hinge to hang it off the wall. Attach a handle to this so you can "lift up" on it. Get

# CONNECT TIME

a relatively large 3-phase motor. We used a surplus 10-horsepower motor. Put a pulley on its shaft. Mount it to the floor below the washing machine motor. Get a big loose belt, put it between the two pulleys. That's the mechanical arrangement.

Wire the washing machine motor so you can plug it in and turn it on and off. Wire the 3-phase motor through a DPDT switch so you can turn it on and off. Wire the respective phases of the motor to the 220 VAC (A phase and B phase). Now, you'll have a third phase left over off the motor. Wire a 3-phase AC connector to the A, B, and C phases. Plug your milling machine into this. You've now constructed a Rotating Transformer Phase shifter.

In the morning, (or whenever) BEFORE you start your milling machine, in the following order, do this: Start the washing machine motor. Pull up on the handle to take up the slack in the belt. This will start your big motor spinning. As it (the big motor) comes up to speed, switch it on, and release the tension on the handle to the washing machine motor. Turn it (the washing machine motor) off. You now have 3-phase 220 VAC being supplied to the milling machine. This will allow you to run your milling machine normally. You will be able to start and stop at will. Try not to start the mill "under load" (cutter buried in some material!) or run the milling machine without the RTP running, as you'll burn it out. At the end of the day, turn off the mill and then the RTP.

Now some caveats. I wish I didn't have to say this, because anyone participating on this BBS has some common sense, but attorneys love this sort of thing. The preceding can be VERY DANGEROUS. If you don't think you can reliably construct it, DON'T. Having an open belt between two motors has the potential to take off a finger or two. The big motor must remain running all the time, and, yes, I have seen visitors walk into the shop, not notice the pulley spinning, and put a foot up on it. The three stooges would have been proud. Also, the voltages present are obviously lethal. Good luck.

**Msg#:42115**

**From: PETE CHOMAK To: MIKE TRIPOLI**

Well, that sounds interesting, but for one Bridgeport with one 1-HP motor, is it really cheaper than just finding a 1-HP single-phase motor to fit the Bridgeport? I imagine that a 10-HP or even a 5-HP three-phase motor will cost a fair amount used, probably more than a 1-HP single-phase will used.

If anyone has any experience with three-phase inverter design, I do have access to some of the 50-amp industrial SCR packs that are used in the servo drives on large CNC machines. Possibly a couple of these could be used to generate the "B" and "C" phases to go with the single

phase? For now I will just drive the neighbors nuts with the diesel generator and look around for a used motor.

**Msg#:42987**

**From: RUSS REISS To: MIKE TRIPOLI**

Mike, I'm intrigued by your idea. But can you explain how you end up with 120-degree phase separation on A, B, C when A and B are hardwired to 220 V with 180-degree difference? What am I missing?

**Msg#:43757**

**From: JAMES MEYER To: RUSS REISS**

Look at it like this:

Even if you have a real three-phase system and you just look at any \*two\* wires, you always see a single phase. Your use of the term "180 degrees" is a bit misleading when you're only talking about two wires. In a two-wire measurement, there \*isn't\* a 180-degree difference.

A residential 220/110-volt system does show a 180-degree phase difference between the two hot legs, but only when you measure them with respect to neutral. And that's a three-wire system.

It's only when you look at more than two wires at the same time that you can see anything other than a single phase no matter what sort of system you have.

If you do manage to get a three-phase motor running on a single-phase supply, and the motor is lightly loaded, the third phase is \*generated\* by the motion of the armature. You've actually got a makeshift motor/generator setup.

**Msg#:43774**

**From: MIKE TRIPOLI To: RUSS REISS**

Jim's right on the money. Using the scheme presented, you get A (0°), B (180°), and C (90°). There are schemes where you can add banks of capacitors and inductors to correct for the phase differences, but the motors used on Bridgeports don't seem to mind. We ran a small shop like this for a couple of years, and never had a problem.

**Msg#:43857**

**From: RUSS REISS To: MIKE TRIPOLI**

OK. I see all that. But the actual "three phase" sent to the motor does not then have 120-degree separation between each phase, right? That's what I was wondering. I have no doubt it can run... just seemed that two of the phases are still locked at 180-degrees apart.

**Msg#:44290**

**From: RUSS REISS To: JAMES MEYER**

Jim, I was thinking more about your reply, and think I grasp it all. You are driving just ONE leg of the three-phase motor with AC (of arbitrary phase). The other TWO phases

# CONNECT TIME

(not just one as a single missing lead might at first imply) are generated by the geometry and rotating magnetics within the motor itself. OK? I got it? Disregard my other note to Mike. His reference to 0, 180, 90 degrees made me think something was still amiss. But I think I'm all clear now.. thanks!

**Msg#:46088**

**From: JAMES MEYER To: RUSS REISS**

Almost..

You are driving TWO legs of the three-phase motor with ONE phase. Since you are using 220 VAC, and since you are \*not\* using the neutral of the 220, you are driving the motor with a single phase.

The confusion comes in when you make the assumption that the phase of one leg of domestic 220 VAC is 180 degrees to the phase of the other leg. That is true \*only\* when you reference the measurement on each leg to neutral. In other words, \*only\* when you have a THREE-wire connection. If you disregard the third leg (the neutral), then you have a TWO-wire connection, and a TWO-wire connection is a SINGLE-phase connection.

This is the way the "rotary transformer" works. The 220-volt house current neutral is NOT used. Lines A and B of the motor are connected to the two 220-VAC hot wires. The third line, C, of the motor then provides the third phase of a three-phase system.

Whenever you use the term "phase," you must always use it in reference to some other value.

*With respect to the house AC neutral* the motor's A leg is 180 degrees ahead or behind the motor's B leg.

However, at the same time, and without being contradictory..

*With respect to the motor's C leg* the motor's A leg is 120 degrees ahead (or behind) and the motor's B leg is 120 degrees behind (or ahead). Ahead or behind depending on which direction the motor is spinning.

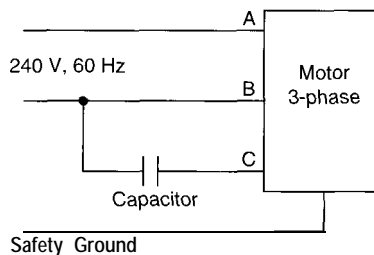
To confuse things even more, if the motor were "Wye" connected, and you measured the phase difference from the 220-volt house neutral to the center point "neutral" of the motor, you would find a 60-degree phase lead or lag. Hint:  $180 - 120 = 60$ .

**Msg#:46574**

**From: PELLERVO KASKINEN To: PETE CHOMAK**

To run a 3-phase motor on a single-phase supply, you need one AC-type capacitor of an appropriate value and voltage rating. This capacitor should produce an adequate phase shift for the third phase winding, while also not dropping too much of the voltage. Given the often high currents at the very low frequency, the capacitances tend to be high.

Here is a sketch of what the connections look like:



It may not be obvious, but by changing the incoming power from the center leg B of the motor windings to the bottom one (C), you get a reversed running direction. In other words, as the capacitor remains between B and C, the supply would be reconnected to the other end of the capacitor.

The difficult issue, I gather, is the sizing of the capacitor. This happens to be one of those questions where there is no explicit answer. The motor is not operating in a linear fashion, especially at very low speeds immediately after the start. You may have heard the howling a motor gives then. Part of that noise is due to the magnetic saturation that both the stator and the rotor experience at this time. Incidentally, do not put anything magnetically sensitive close to an AC motor that is about to start!

Luckily, the AC motor is quite robust. It can be operated with very nonideal supplies, both what concerns the voltage level and the phase. This and the fact that mostly we do not try to run the motor at precisely 100% of its nominal load makes things easier for us.

I took your 1-HP motor, assumed a 230-V supply, and got the following numbers to put the task into perspective: At 230 V, 3-phase, the nominal current would be about 2 A per phase. These of course are nominally 120 degrees apart and add up to zero in the neutral. Therefore, a neutral wire is not required. A safety ground of course is necessary for other reasons. The 3-phase voltages are expressed as between any two phases.

If you have a (center-tapped) 240-V single-phase supply, you are basically home free as far as the voltage goes. No transformer is required. What you need is at least a Starting Capacitor. It is also helpful to have a Running Capacitor, or you can combine the two for a simpler implementation. As it turns out, the value of the Running Capacitor has a definite effect on the power that the motor can produce. The motor runs without a running capacitor, but is probably capable of only 2035 % of the nominal power. With a capacitor you may double that number.

For the starting time, we definitely need a capacitor. It also is higher in value than what normally would be used for a Running Capacitor. Normally, dedicated Starting

# CONNECT TIME

Capacitors are special bipolar electrolytic capacitors for a lower price. But they require some means of shutting off after a start, such as a centrifugal speed switch. Oil paper or polypropylene capacitors can be wired as I show above and remain there, provided they are properly sized.

We only need approximate values, so let's just say we want to select a capacitor that would have the same impedance as the motor winding represents during running. This translates to the same 2-A current when 230 V is applied to the capacitor only. At 60 Hz, this means something on the order of 20 microfarads. An AC capacitor of 15  $\mu$ F, 300 V or more seems suitable. Newark (the only place I checked) carries them for \$28 or so.

I actually ran some circuit analysis to verify this rule of thumb. The selected value appears pretty good for the start time. It is a little high for the run time, but then again, it does not need any speed switches or other complicating hardware.

I hope this answers your question. Ask if you need more details.

## Msg#:54024

From: PETE CHOMAK To: PELLERVO KASKINEN

Thanks for your reply, I dug through my trusty heap and located a couple of 12- $\mu$ F, 660-VAC caps. I hooked one of them up as shown, and tried it out, the results were:

- 1) A nice loud 60-Hz hum
- 2) Little or no movement of the motor
- 3) A manual kick start of the motor via the spindle draw bar would get it turning, VERY SLOWLY, still humming and showing no signs of speeding up.

You mentioned that the motor should draw about 2 A per phase, but the motor nameplate lists it as 4.2 A @ 220 VAC, or 2.1 A @ 440 VAC. I decided to try using both of the caps in parallel thinking that the value you listed might be low by half. The results were:

- 1) A nice softer 60-Hz hum
- 2) The motor would start turning slowly, and not pick up much speed
- 3) A manual kick start would seem to get it over the "hump" and it would accelerate and then settle in at what appears to be the correct speed.
- 4) Some test cutting on a scrap block of steel showed that the motor was producing a reasonable amount of power, with no problems of stalling.

If you have any suggestions on the startup problem let me know, but as it stands now it will work fine (and will get a lot of use!). Thanks again for your help.

## Msg#:54324

From: PELLERVO KASKINEN To: PETE CHOMAK

Interesting! I am not sure what caused me to state the amount of expected current. But whatever it was, the consequences are clear-if the nameplate current is double from my estimate, then the capacitor value has to be doubled.

You might also get one of those start-duty-only capacitors, at least 5 times the run-duty value, and have that through a momentary push button in parallel with the run capacitor. When you turn the power on, you would also push the button for a couple of seconds to get the motor "over the hump." There apparently is too much friction in your system for my suggested values to be on the mark.

## Msg#:54773

From: PETE CHOMAK To: PELLERVO KASKINEN

Interesting, I was thinking of putting a larger cap on with a run/ start momentary switch. There is not much friction in the system, but the starting load will vary depending on what speed the drive is set for. There are 8 speeds, with a 4-step pulley and a 2-stage gear drive. A starting cap will help, as it is difficult to get it going manually at the extremes of the speed range. Thanks.

---

*We invite you 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 (203) 871-1988. Set your modem for 8 data bits, 1 stop bit, no parity, and 300, 1200, 2400, 9600, or 14.4k bps. For information on obtaining article software through the Internet, send E-mail to [info@circellar.com](mailto:info@circellar.com).*

## ARTICLE SOFTWARE

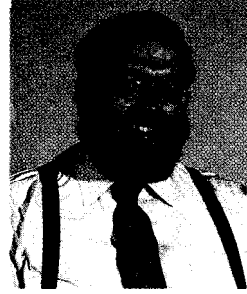
Software for the articles in this and past issues of *The Computer Applications Journal* may be downloaded from the Circuit Cellar BBS free of charge. For those unable to download files, the software is also available on one 360 KB IBM PC-format disk for only \$12.

To order Software on Disk, send check or money order to: The Computer Applications Journal, Software On Disk, P.O. Box 772, Vernon, CT 06066, or use your VISA or Mastercard and call (203) 8752199. Be sure to specify the issue number of each disk you order. Please add \$3 for shipping outside the U.S.

428 Very Useful    429 Moderately Useful    430 Not Useful

# STEVE'S OWN INK

Once Every 27,000 Years



O

here are good and bad things about having a BBS. The good news is that it is a fast and convenient communication vehicle. The bad news is the same. The sheer volume of mail is overwhelming. In fact, the worse situation for me is sitting down to read all the Internet stuff right after a large lunch...

"Steve, we have a problem with one of the program listings in the latest issue," Ken emphatically states as he paces back and forth. "Our audience counts on our accuracy. This isn't just any typo! Anyone entering and running this program occasionally gets a rounding-off error out past the seventh decimal place!"

Wiping his brow, Ken contemplates the mass hysteria a seventh-digit rounding error could mean to a die hard Circuit Cellar *INK* reader. Scandalous, at the very least.

I could feel a rising queasiness in my stomach as Lionel, the corporate bean counter, and Janice come rushing in. Breaking into our discussion with an urgency rarely observed, Janice pronounces, "It's all over the Internet! A math professor in Virginia entered our program listing and claims that it might be in error! If it's true, he wants a new magazine with the listing corrected!"

I cringe at the thought of spending \$100k to reprint and mail an issue, but...

Holding to financial reason, Lionel rebuts, "Wait, I looked at the problem and don't see that we have to do anything!" (I'll never understand why they call these bean counters "money people." Money always seems to be the last thing they're willing to part with.)

Again he opines, "Look, an issue has more than a ¼ million characters, very few readers will notice a simple typo. Fewer still will really use the program. If I had to extrapolate the odds, let's see..." It was as though his mind was one big spreadsheet. His eyes glistened as he pounded on the pocket portable computer that he fondled like others do their pocket knife.

Suddenly, he looked up, satisfied with his results. "I calculate that the ordinary reader has a 9 billion to one chance of a problem. Heck, what's that? Once in 27,000 years? Let's just forget about it!"

But, as the latest Internet news spewed from the printer, Janice erupts, "Some guys at IBM have entered the program and are telling everyone else. Seems to me that if everyone is told about the error, then 9 billion to one degrades to about one in one."

Even in such a critical situation as this, I could detect Janice's exhilaration at getting the upper hand on the bean counter.

He instantly quips, "So what! OK, we'll make some corrected copies, but I want the reader to swear he really needs better than seventh digit accuracy before we send it to him!"

My heart raced as I contemplated this debacle. Asking readers to justify their use would be a public-relations disaster. That was out of the question. Equally unappealing was going to the publishers office and suggesting to Dan that we might want to reprint 50,000 magazines to correct a seventh digit round-off error.

Why couldn't I have been in the software business instead. All their errors and **disfunctions** simply create a need for the next software revision skillfully marketed as an "upgrade."

"How about our next issue being called an upgrade..."

"Steve.. Steve.. ..!"

Embarrassed at being aroused from an apparently sound sleep at my computer, I expressed a humiliated smile at Dan who stood shaking my shoulder.

"The blues are in the conference room-last chance to proof the issue before it's printed."

Asleep or not, my heart was still pounding from my "near typo experience." I jumped up immediately and emphatically responded, "You better believe it!"