#196 November 2006

www.circuitcellar.com

CIRCUIT CELLAR®

THE MAGAZINE FOR COMPUTER APPLICATIONS

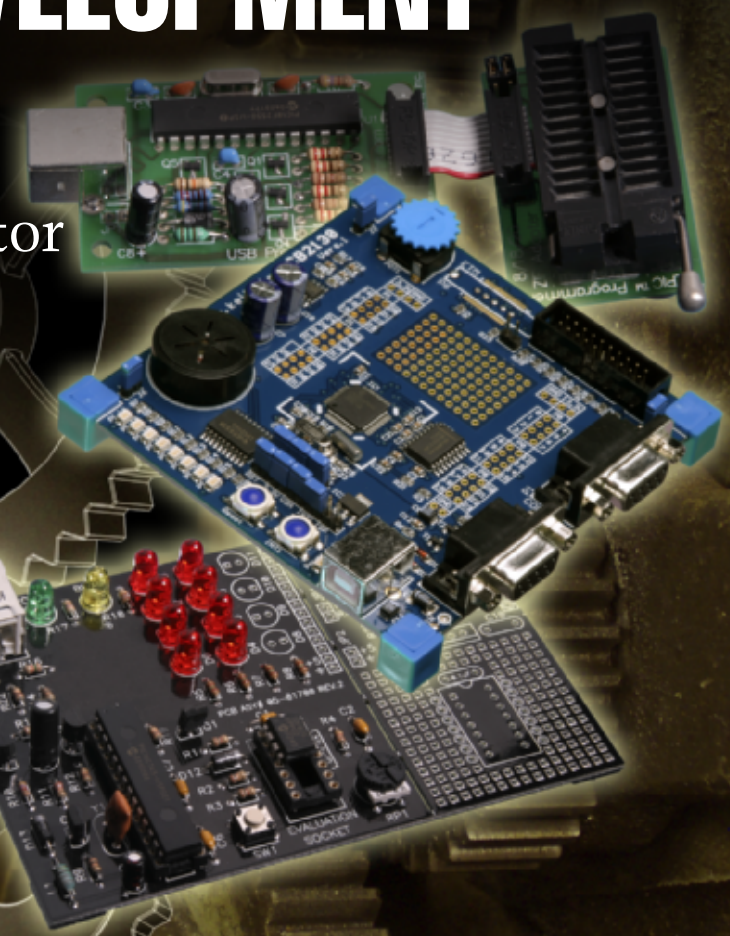
EMBEDDED DEVELOPMENT

MCU-Based Probing

Build a Gas Leak Detector

Modular Physical User Interface System

Changes in Memory Technology



11>
7 25274 75349 9
$4.95 U.S. ($5.95 Canada)

32-bit performance at an 8-bit price

Introducing the NetBurner Mod5213

for embedded applications



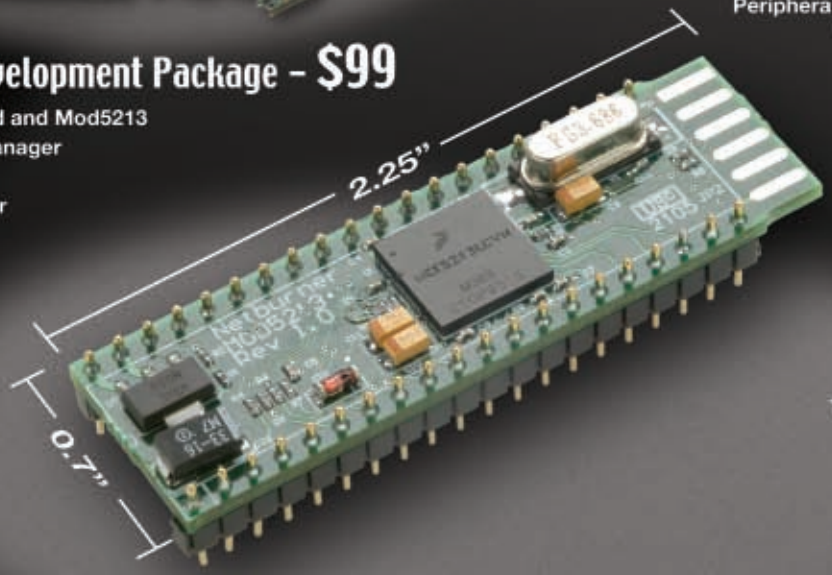
Mod5213 – $39*

Processor	Freescale 32-bit MCF5213
Speed	66MHz, 62 MIPS
Flash Memory	256 KB
SRAM	32 KB
Size	2.25" x 0.7" 40-pin DIP
Peripherals	Up to 33 GPIO 8-channel 12-bit Analog to Digital
	Serial Interfaces 3 UARTs DMA capable SPI I2C CAN 2.0
	Timers Four 32-bit timer channels with DMA capability
	Four 16-bit timer channels with capture/compare/PWM
	4-channel 16-bit/8-channel 8-bit PWM generator
	Two periodic interrupt timers (PITs)
	Special Features Hardware accelerated Multiply, Accumulate and Divide
	4-channel DMA controller
	Watchdog Timer
Electrical Specs	Input Voltage 4 -7 Volt
	Stop mode Current 130uA
Temperature range	-40°C to +85°C

*quantity 1,000

Complete Development Package – $99

- Development Board and Mod5213
- IDE with project manager
- C/C++ Compiler
- Graphical Debugger
- Real-Time OS
- Deployment Tools



NetBurner

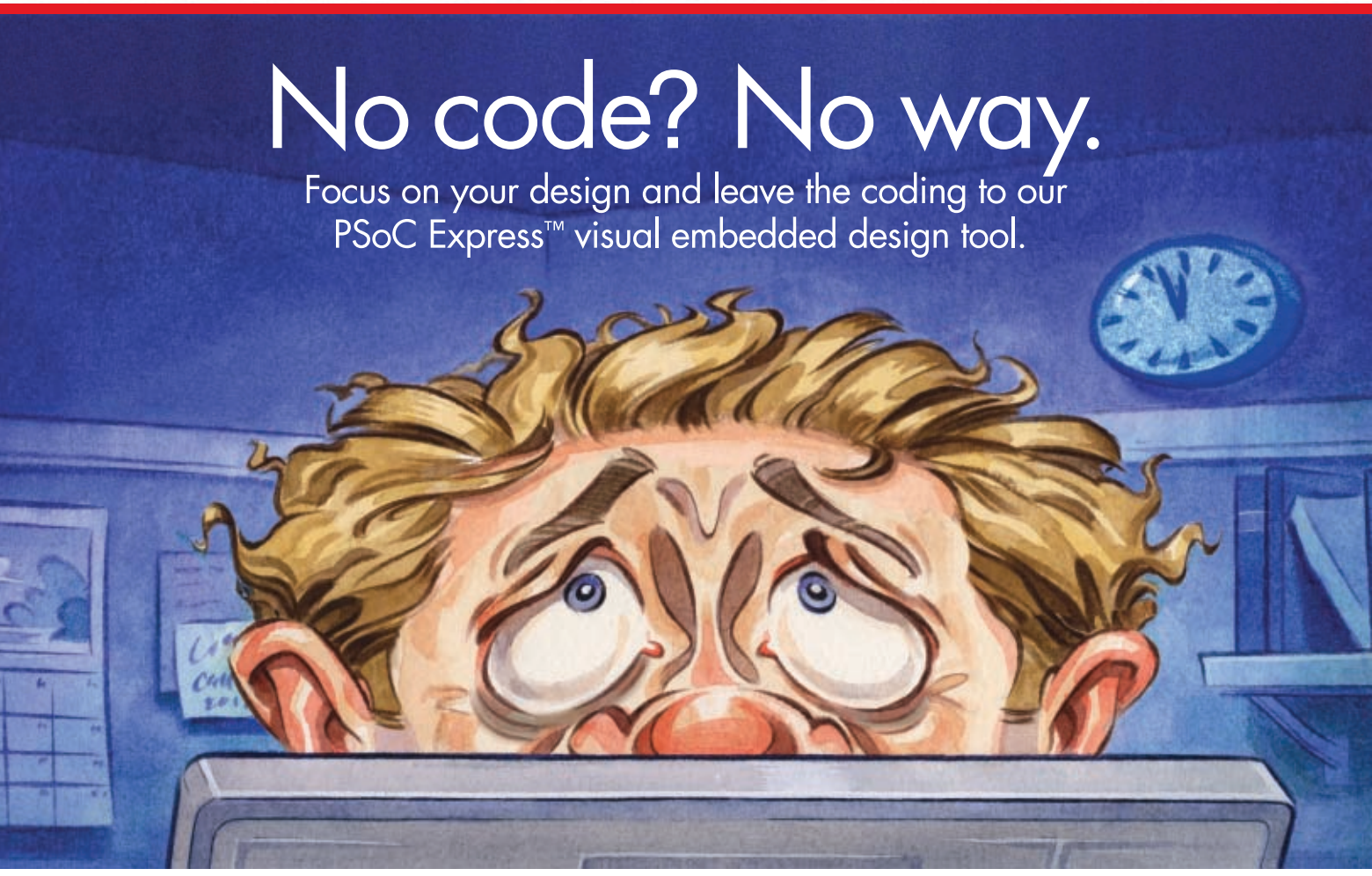
Networking in 1 Day!

www.netburner.com • 1-800-695-6828

freescale semiconductor

No code? No way.

Focus on your design and leave the coding to our PSoC Express™ visual embedded design tool.



Without writing a single line of assembly or "C" code, generate a complete custom design with PSoC Express. Here's what you get with Version 2.1 of our visual embedded software:

- *Support for our family of Programmable System-on-Chip™ (PSoC®) mixed-signal arrays—powerful, programmable digital and analog blocks with integrated MCU and flash memory*
- *Rich visual environment with simulation enables you to see your design and evaluate its performance instantly*
- *Built-in support for interdevice communication; seamlessly divide design problems into smaller pieces using multiple PSoC devices*
- *Retarget to any PSoC mixed-signal array at any time; design first and select device later*

TRY IT OUT NOW.

Whether you have a minute or an hour to invest, we have a way to get you working with PSoC Express:

- *View our online PSoC Express demo:* **www.cypress.com/expressdemo**
- *Request a FREE PSoC Express evaluation kit:* **www.cypress.com/expresskit**
- *Test drive PSoC Express at a live seminar near you:* **www.cypress.com/expresstour**
- *Download FREE PSoC Express software:* **www.cypress.com/psocexpress**

Make all your embedded designs fast and easy.
www.cypress.com/getexpress

CYPRESS
PERFORM

Cypress, the Cypress logo and PSoC are registered trademarks, and Programmable System-on-Chip and PSoC Express are trademarks of Cypress Semiconductor Corporation. All other trademarks are properties of their respective owners. ®2006 Cypress Semiconductor Corporation. All rights reserved.

How far will your design take you?







DESIGN STELLARIS 2006

Challenge yourself against other top embedded engineers around the world in **DesignStellaris2006**, proudly sponsored by Luminary Micro, Keil, and Circuit Cellar.

Use any microcontroller in Luminary Micro's Stellaris™ family of ARM® Cortex™- M3 controllers with the ARM RealView® Microcontroller Development Kit (MDK-ARM) to create your design contest entry, and see how far your design will take you!

- No purchase necessary to enter.
- $10,000 in cash prizes!
- Entry deadline is February 7, 2007.
- Winners will be announced at the Embedded Systems show... Silicon Valley 2007.
- Submit your design today!

FOR COMPLETE DETAILS, VISIT
www.LuminaryMicro.com/DesignStellaris2006

DESIGN STELLARIS 2006

LUMINARY MICRO™

KEIL™ An ARM® Company

Cortex™ Intelligent Processors by ARM®

CIRCUIT CELLAR®

www.LuminaryMicro.com/DesignStellaris2006

LUMINARY MICRO LM3S811 EVALUATION KIT

Visit our website for more information!



THE LM3S811 EVALUATION KIT includes
the Stellaris LM3S811 Evaluation Board, an evaluation copy of MDK-ARM, USB cable, documentation, and programming examples.

PLC / Embedded controller

CUBLOC



MODEL	CB220	CB280	CB290
FLASH	80KB	80KB	80KB
I/O	16	49	91
DATA	3KB	3KB	28KB
PRICE	$34	$43	$69

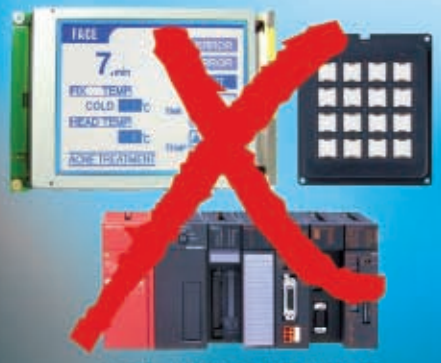
CUBLOC™ (kyoob-lŏk) ***n.*** 1. Easy-to-use Industrial Controller capable of running both BASIC and Ladder Logic. Simply add buttons, sensors, motors, relays, and an LCD to finish project on-time.
2. Could-this-be-true? **μ**-sized **B**asic and **L**adder **O**n **C**hip?

- Use BASIC with Ladder Logic language
- Pin compatible with BS2 (CB220 only)
- 80KB Flash User Program memory
- 2~24KB Data memory
- BASIC Speed : 36000 inst./Sec
- RS232, SPI, I2C, MODBUS supported
- Share Data memory between BASIC & Ladder Logic
- All Development Software Included
- Technical support to jump start your project

Consolidate your GUI before it's too LATE!

ALL-IN-ONE Type Touch Panel Controller

CUTOUCH



LCD + KEYPAD + PLC



COMFILE TECHNOLOGY

550 Pilgrim Drive Suite K Foster City, CA 94404
call : 1-888-9CUBLOC (toll free)
1-888-928-2562

* 14-Day Money-Back Guarantee

Log on to
www.comfiletech.com
for your FREE catalog!

TASK MANAGER

Making a Contribution

Each of us is all the sums he has not counted: subtract us into nakedness and night again, and you shall see begin in Crete four thousand years ago the love that ended yesterday in Texas ... The seed of our destruction will blossom in the desert, the alexin of our cure grows by a mountain rock, and our lives are haunted by a Georgia slattern, because a London cutpurse went unhung. Each moment is the fruit of forty thousand years.

—Thomas Wolfe, *Look Homeward, Angel* (Charles Scribner's Sons, 1929)

So too is each new embedded design the fruit of hundreds if not thousands of other designs.

I read Thomas Wolfe's remarkable first novel *Look Homeward, Angel* at least once a year. When I was reading the first page a few weeks ago, I began thinking about the technologies and projects covered in *Circuit Cellar*. In the beginning paragraphs of the book, Wolfe poetically describes the causal relationships between past and present occurrences, ideas, and situations. For instance, he suggests that a chain of events that started in Crete 4,000 years ago ended yesterday in Texas. How does this idea relate to the projects described in *Circuit Cellar* every month? Well, each design has a particular past, present, and future that's part of a much longer story of technological progress. For instance, Alexander Enzmann's USB flash PIC programmer (p. 12) didn't suddenly appear out of thin air. Literally thousands of other designs had to be tested and tweaked before Alexander could build his programmer. Thus, his design is the result of years of technological development and idea swapping between engineers.

This is an interesting perspective from which to view projects described in this magazine. Rather than looking at a project as simply both a start and an end point, you should take a step back and look at the technological developments that led up to the particular design. After doing so, you can then consider how the design and the principles behind it might contribute to your future projects or the designs of others. By doing so, you'll see that every design we feature, regardless of its complexity, plays an important role in the progress of embedded development in the early 21st century.

What does this mean for you? Well, you may not need a gas leak detector, but by reading Brian Millier's article (p. 44) and studying his schematics and code, a seed will be sown in your mind that will surely reap a reward somewhere down the line, whether you end up incorporating an aspect of the design in a ground-breaking technology or simply referring the project to a fellow engineer in need of a little assistance. The idea is to recognize that each design is a small contribution to the ever-evolving process of technological development.

As you continue reading *Circuit Cellar* and working on your own designs, consider making a contribution of your own. If you send me an article proposal (cj@circuitcellar.com), we will review it and respond in fairly short order. It doesn't get much easier than that!

Recently, we've seen many interesting proposals by designers who are dedicated to developing the new technologies that will change the way we live and interact with each other and our environment in this new century. This type of proactive approach to tackling the major problems of our day is what we like to see. Projects that address solutions to problems associated with topics such as energy consumption and emissions/EMI monitoring have everyone excited. I encourage you to get serious about an important issue and then start designing a solution. You can then use *Cellar Cellar* to bring your ideas to the masses.

cj@circuitcellar.com

CIRCUIT CELLAR®

THE MAGAZINE FOR COMPUTER APPLICATIONS

FOUNDER/EDITORIAL DIRECTOR
Steve Ciarcia

MANAGING EDITOR
C.J. Abate

WEST COAST EDITOR
Tom Cantrell

CONTRIBUTING EDITORS
Jeff Bachiochi
Ingo Cyliax
Fred Eady
George Martin
Ed Nisley

NEW PRODUCTS EDITOR
John Gorsky

PROJECT EDITORS
Steve Bedford
Ken Davidson
David Tweed

CHIEF FINANCIAL OFFICER
Jeannette Ciarcia

MEDIA CONSULTANT
Dan Rodrigues

CUSTOMER SERVICE
Debbie Lavoie

CONTROLLER
Jeff Yanco

ART DIRECTOR
KC Prescott

GRAPHIC DESIGNER
Mary Turek

STAFF ENGINEER
John Gorsky

ADVERTISING

860.875.2199 • Fax: 860.871.0411 • www.circuitcellar.com/advertise

PUBLISHER
Sean Donnelly
Direct: 860.872.3064, Cell: 860.930.4326, E-mail: sean@circuitcellar.com

ADVERTISING REPRESENTATIVE
Shannon Barraclough
Direct: 860.872.3064, E-mail: shannon@circuitcellar.com

ADVERTISING COORDINATOR
Valerie Luster
E-mail: val.luster@circuitcellar.com

Cover photography by Chris Rakoczy—Rakoczy Photography
www.rakoczyphoto.com

PRINTED IN THE UNITED STATES

CONTACTS

SUBSCRIPTIONS
Information: www.circuitcellar.com/subscribe, E-mail: subscribe@circuitcellar.com
Subscribe: 800.269.6301, www.circuitcellar.com/subscribe, Circuit Cellar Subscriptions, P.O. Box 5650, Hanover, NH 03755-5650
Address Changes/Problems: E-mail: subscribe@circuitcellar.com

GENERAL INFORMATION
860.875.2199, Fax: 860.871.0411, E-mail: info@circuitcellar.com
Editorial Office: Editor, Circuit Cellar, 4 Park St., Vernon, CT 06066, E-mail: editor@circuitcellar.com
New Products: New Products, Circuit Cellar, 4 Park St., Vernon, CT 06066, E-mail: newproducts@circuitcellar.com

AUTHORIZED REPRINTS INFORMATION
860.875.2199, E-mail: reprints@circuitcellar.com

AUTHORS
Authors' e-mail addresses (when available) are included at the end of each article.

CIRCUIT CELLAR®, THE MAGAZINE FOR COMPUTER APPLICATIONS (ISSN 1528-0608) is published monthly by Circuit Cellar Incorporated, 4 Park Street, Vernon, CT 06066. Periodical rates paid at Vernon, CT and additional offices. **One-year (12 issues) subscription rate USA and possessions $23.95, Canada/Mexico $34.95, all other countries $49.95. Two-year (24 issues) subscription rate USA and possessions $43.95, Canada/Mexico $59.95, all other countries $85.** All subscription orders payable in U.S. funds only via Visa, MasterCard, international postal money order, or check drawn on U.S. bank. **Direct subscription orders and subscription-related questions to Circuit Cellar Subscriptions, P.O. Box 5650, Hanover, NH 03755-5650 or call 800.269.6301.**

Postmaster: Send address changes to Circuit Cellar, Circulation Dept., P.O. Box 5650, Hanover, NH 03755-5650.

Circuit Cellar® 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® disclaims any responsibility for the safe and proper function of reader-assembled projects based upon or from plans, descriptions, or information published by Circuit Cellar®.

The information provided by Circuit Cellar® is for educational purposes. Circuit Cellar® makes no claims or warrants that readers have a right to build things based upon these ideas under patent or other relevant intellectual property law in their jurisdiction, or that readers have a right to construct or operate any of the devices described herein under the relevant patent or other intellectual property law of the reader's jurisdiction. The reader assumes any risk of infringement liability for constructing or operating such devices.

Entire contents copyright © 2006 by Circuit Cellar, Incorporated. All rights reserved. Circuit Cellar is a registered trademark of Circuit Cellar, Inc. Reproduction of this publication in whole or in part without written consent from Circuit Cellar Inc. is prohibited.

MC1321x ZigBee™ Family



Low power 2.4GHz radio frequency transceiver and 8-bit microcontroller in a 9x9x1mm 71-pin LGA package. Provides up to 60KB of Flash memory and 4KB of RAM. Supports star, mesh, and cluster tree networks.

freescale semiconductor mouser.com/freescale/a

DLP-RFID1 Module



Read/write to ISO 15693, ISO 18000-3, and Tag-It™ intelligent RFID transponder tags. USB 1.1/2.0 compatible, built-in antenna with up to a 4-in. read range, 13.56MHz, reads/writes up to 256 bytes of data and up to 15 UID/SID tags simultaneously.

DLP Design mouser.com/dlp/a

The Mouser Advantage: Faster Time to Market for YOUR New Designs!



NEWEST Products
NEWEST Technologies
The ONLY NEW Catalog Every 90 Days

For over 40 years engineers have relied on Mouser as their source for electronic components. And with the most rapid introduction of new products, Mouser gives you a critical time-to-market advantage.

That's why we deliver the **ONLY** 1,800+ page catalog of the NEWEST product information **4 times a year**. And with daily updates to over 700,000 products on-line, you can depend on Mouser to save you critical time to market!

Experience Mouser's time-to-market advantage! Our vast selection of the NEWEST products, NEWEST technologies, no minimums, and same-day shipping on most orders, gets you to market faster. We make it easy to do business with Mouser!

mouser.com **(800) 346-6873**

MOUSER ELECTRONICS
a tti company

The Newest Products For Your Newest Designs

SAS/SATA iPass™ Connectors



Multi-Lane and Mini Multi-Lane high-speed serial interconnects provide internal and external storage solutions to SAS and SATA. For applications ranging from 1.5 to 6.0 Gbps, with capability up to 10Gbps. Serial technology uses smaller cables for bandwidth-intensive applications.

molex one company › a world of innovation mouser.com/molex/a

RF and Microwave Coaxial Switches



CCR-33 and CR-33Series of failsafe, latching, and multi-throw coaxial switches. Operates from DC-18GHZ or DC-22Ghz, 50 ohms, broadband, SPDT, electromechanical. Designed to switch a microwave signal from a common input to either of two outputs.

TELEDYNE RELAYS A Teledyne Technologies Company mouser.com/teledyne/a

Multilayer Chip Inductors – LK Series



Internal printed coil structure creates a closed magnetic circuit that eliminates crosstalk and allows for higher mounting densities. Rated current 25mA, DC resistance 1.50, temp. range -40°C to +85°C, self-resistant frequency 120MHz.

KEMET CHARGED. mouser.com/kemet/a

The NEWEST Semiconductors | Passives | Interconnects | Power | Electromechanical | Test, Tools & Supplies from Mouser Electronics

Mouser and Mouser Electronics are registered trademarks of Mouser Electronics, Inc. Other products, logos, and company names mentioned herein, may be trademarks of their respective owners.

November 2006: Embedded Development

FEATURES

12 **USB Flash PIC Programmer**
Alexander Enzmann

20 **Control Panel Design**
Implement a Modular Physical User Interface System
Eric Gagnon

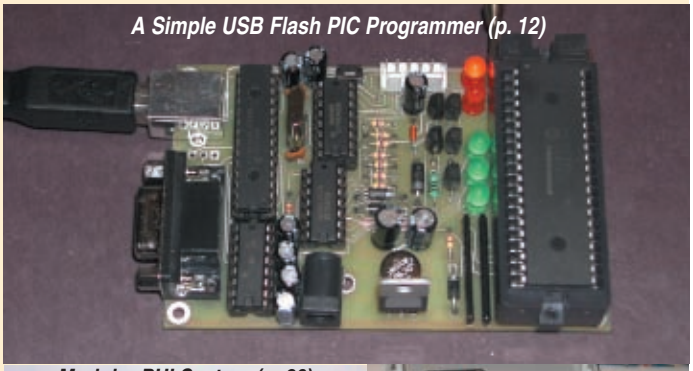
32 **Open-Source AVR Development**
Colin O'Flynn

36 **Electronic Gear Control**
Add Electronic Gears to a Metal Lathe
John Dammeyer

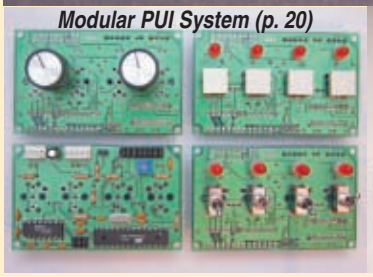
44 **Low-Cost Gas Leak Detector**
Brian Millier

52 **TV-Based Oscilloscope**
Ilya Mamontov

58 **Smart Probing**
An R5F2113-Based Soil-Monitoring System
Alain d'Espaignet



A Simple USB Flash PIC Programmer (p. 12)



Modular PUI System (p. 20)



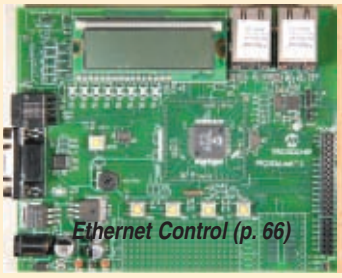
Gas Leak Detection (p. 44)

COLUMNS

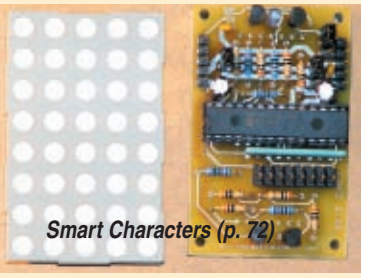
66 **APPLIED PCs**
Stand-Alone Ethernet Control
Fred Eady

72 **FROM THE BENCH**
Smart Character Display
MCU-Based Information Messaging Made Simple
Jeff Bachiochi

78 **SILICON UPDATE**
Memory Lane Change
Tom Cantrell



Ethernet Control (p. 66)



Smart Characters (p. 72)

DEPARTMENTS

4 **TASK MANAGER**
Making a Contribution
C.J. Abate

8 **NEW PRODUCT NEWS**
edited by *John Gorsky*

93 **CROSSWORD**

94 **INDEX OF ADVERTISERS**
December Preview

96 **PRIORITY INTERRUPT**
In Search of the Real Thing
Steve Ciarcia



AVR picoPower Microcontrollers

To meet the tough requirements to modern microcontrollers Atmel® has now combined ten years of low power research and development into picoPower™ technology for AVR® microcontrollers. picoPower enables AVR to achieve the industry's lowest power consumption with 650 nA with a real time counter running and 100 nA in deep sleep.

What can AVR picoPower do for your design?

- True 1.8V supply voltage enabling operation of all features and core down to 1.8V
- Minimized leakage current enabling 100 nA Power Down sleep consumption
- Sleeping brown-out detector enabling full protection with no power penalty
- Ultra low power 32 kHz crystal oscillator enabling operation at only 650 nA

For more information, check out **www.atmel.com/ad/picopower**

© 2006 Atmel Corporation. All rights reserved. Atmel®, logo and combinations thereof, AVR®, Everywhere You Are® are registered trademarks of Atmel Corporation or its subsidiaries. Other terms and product names may be trademarks of others. All Characters in this document are created by Mykle and Fantasi-Fabrikken AS 2006.

Everywhere You Are®

NEW PRODUCT NEWS

Edited by John Gorsky

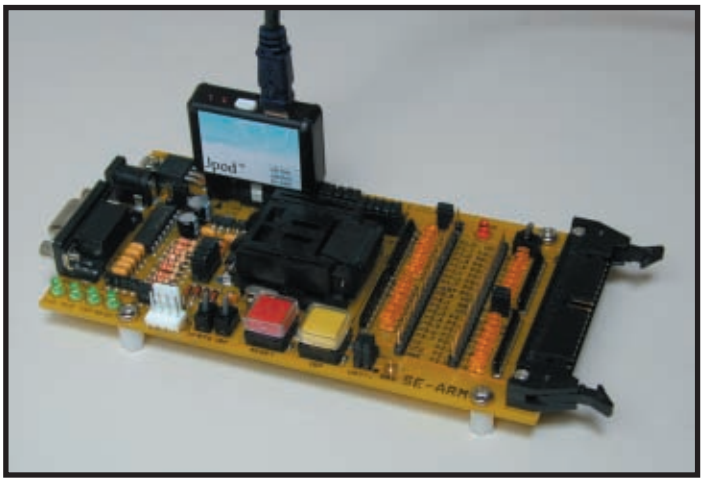
STARTER KIT FOR LPC210X ARM-BASED CHIPS

The **Philips LPC210X Starter Kit** provides developers with a highly flexible and feature-rich environment for rapid program development for LPC210X ARM-based microcontrollers. The kit also can be used to drive legacy 8051 boards.

The kit supports all of the LPC210X family of MCUs, including the older LPC2104/5/6 chips and the more recent LPC2101/2/3 chips. The kit includes an SE-ARM-P target board fitted with an LPC2103 chip. The clamshell socket that houses the MCU enables the chip to be readily replaced by any of the other MCUs in the family, thus making the board useful as an LPC210X programming tool.

A 32-KB version of Crossware's ARM Development Suite software is also included. This enables you to create full working programs in seconds using the default start-up files created by the Crossware environment and the fully integrated Code Creation Wizards. Programs can be downloaded to the board using the Philips flash utility or alternatively via the Jaguar USB debugging interface, which facilitates super-fast download via the JTAG port.

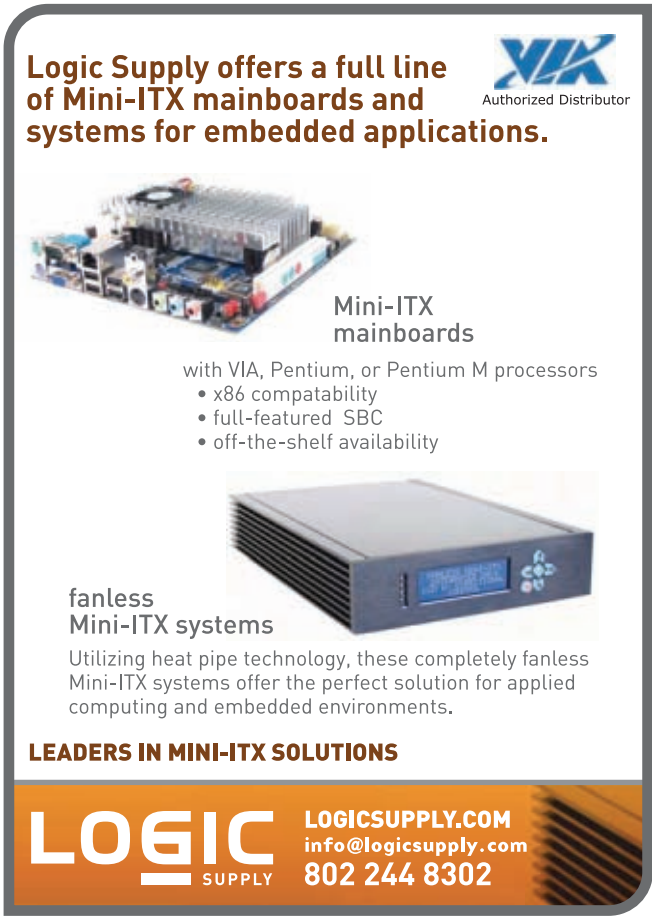
The evaluation board has been constructed so that it is able to drive legacy 8051 boards. A ribbon cable connected to the starter kit board plugs into the socket normally occupied by the 8051. The ARM chip drives the 8051 board itself. A switch enables the 8051 board to be driven at either 3.3 or 5 V. This feature enables you to rapidly assess the potential of changing from an 8051-based chip to an LPC210X ARM-based MCU.



The starter kit costs approximately **$280**. The Jaguar interface costs approximately **$470**.

Crossware
www.crossware.com





NEW PRODUCT NEWS

STELLARIS LM3S102 EVALUATION KIT

A new version of the **KickStart kit** supports the Luminary Micro LM3S102, a member of the Luminary Stellaris family of microcontrollers, which features the ARM Cortex-M3 32-bit microcontroller core. The KickStart kit provides a low-cost route for MCU system developers to test drive the Stellaris microcontrollers. It includes a free tool suite that enables you to progress rapidly to a working system. The ground-breaking ARM Cortex M3 architecture featured in Stellaris microcontrollers provides 32-bit performance with an ease of use and price point that rival those of 8-/16-bit devices.

Each KickStart kit for the LM3S102 includes an evaluation board fitted with the Luminary microcontroller and an applications CD-ROM containing IAR Embedded Workbench V. 4.40 for ARM with Cortex-M3 support in an 8 KB code-size limited version that supports the full flash capability of the LM3S102. The kit includes a debugger with flash memory download capability, an integrated IAR J-Link JTAG interface, a USB cable, a QuickStart Guide, and documentation and warranty information for both the IAR Embedded Workbench and the Luminary Micro Stellaris microcontroller.

The integration of the stable IAR J-Link JTAG interface technology onto the board provides an innovative and highly usable means of rapidly and effectively connecting to the host for debugging and flash memory download. IAR J-Link integrates seamlessly into IAR Embedded Workbench and is fully plug-and-play compatible.



The KickStart kit for the LM3S102 is now available for **$99** at www.luminarymicro.com.

IAR Systems, Inc.
www.iar.com

EMBEDDED USB HOST CONTROLLER DEVICES

The new **Vinculum** USB host controller ICs handle the USB host interface and data transfer functions. Owing to the built-in 8-/32-bit MCU and embedded flash memory, Vinculum encapsulates the USB device classes as well. When interfacing to mass storage devices such as USB flash drives, Vinculum also transparently handles the FAT file structure communicating via UART, SPI, or parallel FIFO interfaces via an easy-to-implement command set.

The initial product member of the family is the VNC1L device, which features two USB ports that can be individually configured by firmware as host or slave ports. The VNC1L's key features include dual DMA controllers for hardware acceleration, 64-KB flash program memory, and 4-KB data SRAM. As for ports, the ICs offer two USB ports, UART, SPI and parallel FIFO interfaces, and PS2 legacy keyboard and mouse interfaces. Up to 28 GPIO pins are also available depending on the configuration. The device operates at 3.3 V with 5-V safe inputs. Power requirements are only 25 mA when running and 2 mA at standby.



The device is available in an LQFP-64 ROHS-compliant package and costs **$5** in 10,000-piece quantities.

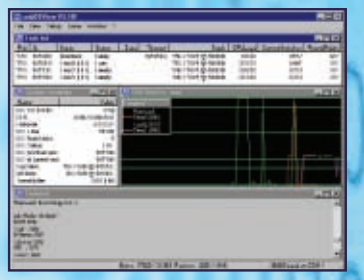
Future Technology Devices International
www.ftdichip.com

+++ NO ROYALTIES +++

embeddedsoftware solutions

Eval versions available

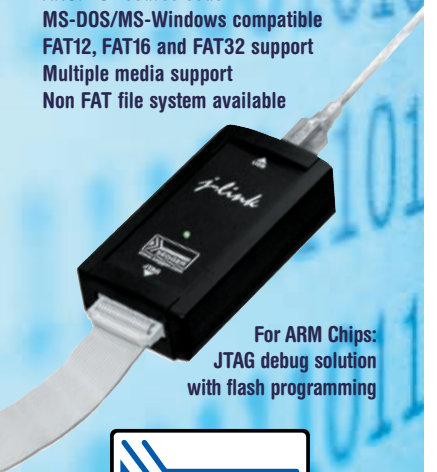
embOS® (RTOS)
+++ 8/16/32 bits +++
Preemptive multitasking
Zero interrupt latency
Easy to use start project included
Profiling support included
Object/source code available



emWin® (GUI)
+++ 8/16/32 bits +++
ANSI "C" source code, no C++ required
Supports b/w, grayscale and color
2D graphic library included
Variety of fonts included
PC simulation included
Window Manager/Widgets (opt)



emFile (File system)
+++ 8/16/32 bits +++
ANSI "C" source code
MS-DOS/MS-Windows compatible
FAT12, FAT16 and FAT32 support
Multiple media support
Non FAT file system available



For ARM Chips:
JTAG debug solution
with flash programming

SEGGER

phone: 978-874-0299
www.segger.com

You can start with U CAN

The unique fully functional 16-bit microcontroller CAN Evaluation Tool



Smart, easy to use and excellent value

Key Features of the XC164CM U CAN START KIT:

- High-performance, real-time 16/32-bit MCU
- On-chip TwinCAN
- Single Cycle Instruction Execution
- 64 kBytes embedded Flash
- DSP support
- Enhanced power-saving modes



THE XC164CM U CAN START KIT is the only low-cost tool available with which you can test and evaluate not only the real-time embedded 16-bit XC164CM microcontroller but also your CAN program. It is a fully enabled CAN evaluation platform with a full-featured HITEX HiTop debugger and Tasking XC166 compiler. The TwinCAN feature of the XC164CM includes 2 CAN transceivers, allowing both CAN nodes to be accessed through a 10-pin header along with analog and digital I/O signals.

Simply plug the start kit into a USB port and install the GUI-driven CAN application program. Or U CAN verify and test your own CAN applications.

You can start with U CAN.

Order now at:

www.infineon.com/us-ucan

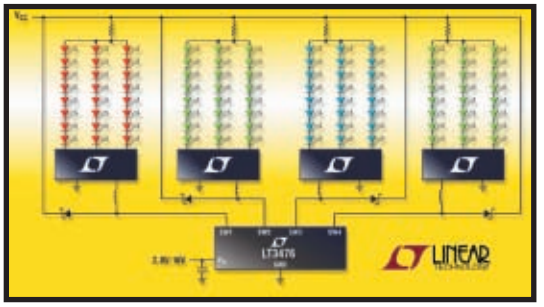
Never stop thinking

NEW PRODUCT NEWS

Visit www.circuitcellar.com/npn for more New Product News.

QUAD LED DRIVER

The **LT3476** is a 2-MHz DC/DC converter designed to operate as a four-channel constant current LED driver. Each of the LT3476's channels can drive up to eight 1-A LEDs in series. This enables the LT3476 to drive up to 32 1-A LEDs while delivering efficiencies up to 96%. Each of the four channels is operated by an independent True Color PWM signal, enabling each channel to be dimmed independently to ratios as high as 1000:1. A fixed-frequency current mode architecture ensures stable operation over a wide range of supply and output voltages. A frequency adjust pin enables you to program the frequency between 200 kHz and 2 MHz to optimize efficiency while minimizing the external component size. Its thermally enhanced 5 mm × 7 mm QFN package provides a highly compact solution footprint for 100-W LED applications.

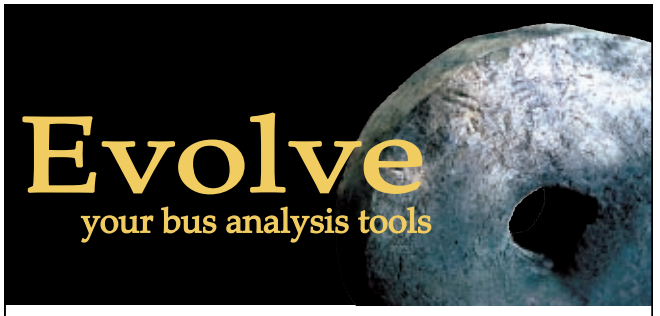


The LT3476 senses output current at the high side of the LED, enabling buck, buck-boost, or boost configurations. With an external sense resistor, you program the output current range of each channel. Each of the four independent driver channels uses an internal 1.5-A, 36-V NPN switch. Other features include open LED protection and thermal limiting.

The LT3476EUHF is available in a thermally enhanced 38-lead 5 mm × 7 mm DFN package. One thousand-piece pricing starts at **$4.64** each.

Linear Technology Corp.
www.linear.com

STATEMENT REQUIRED BY THE ACT OF AUGUST 12, 1970, TITLE 39, UNITED STATES CODE SHOWING THE OWNERSHIP, MANAGEMENT AND CIRCULATION OF CIRCUIT CELLAR,THE MAGAZINE FOR COMPUTER APPLICATIONS, published monthly at 4 Park Street, Vernon, CT 06066. Annual subscription price is $23.95. The names and addresses of the Publisher, Editorial Director, and Managing Editor are: Publisher, Sean Donnelly, 4 Park Street, Vernon, CT 06066; Editorial Director, Steven Ciarcia, 4 Park Street, Vernon, CT 06066; Managing Editor, Carmine J. Abate, 4 Park Street, Vernon, CT 06066. The owner is Circuit Cellar, Inc., Vernon, CT 06066. The names and addresses of stockholders holding one percent or more of the total amount of stock are: Steven Ciarcia, 4 Park Street, Vernon, CT 06066. The average number of copies of each issue during the preceding twelve months is: A) Total number of copies printed (net press run) 23,496; B) Paid/Requested Circulation (1) Mail subscriptions: 9,777; (3) Sales through dealers and carriers, street vendors and counter sales: 5,436 C) Total paid circulation: 15,212; D) Free distribution by mail (samples, and other free issues): 2,472; E)Free distribution outside the mail (carrier, or other means): 1,779; F) Total free distribution: 4,251; G) Total Distribution: 19,463; H) Copies not distributed:(1) Office use leftover, estimated newsstand returns, spoiled after printing: 4,033; I) Total: 23,496. Percent paid and/or requested circulation: 78.2%. Actual number of copies of the single issue published nearest to filing date is October 2006, Issue #195; A) Total number of copies printed (net press run) 25,885; B) Paid/Requested Circulation (1)Mail subscriptions: 9,968; (3) Sales through dealers and carriers, street vendors and counter sales: 5,345; C) Total paid circulation: 15,313; D) Free distribution by mail (samples, and other free issues): 3,790; E) Free distribution outside the mail (carrier, or other means): 2,965; F) Total free distribution: 6,755; G) Total Distribution: 22,068; H) Copies not distributed: (1) Office use leftover, estimated newsstand returns, spoiled after printing: 3,817; I) Total: 25,885. Percent paid and/or requested circulation: 69.4%. I certify that the statements made by me above are correct and complete. Sean Donnelly, Publisher.



Evolve your current bus analysis tools with ATI's CANLab 2006.

Concentrate on analyzing data rather than learning the tool itself.

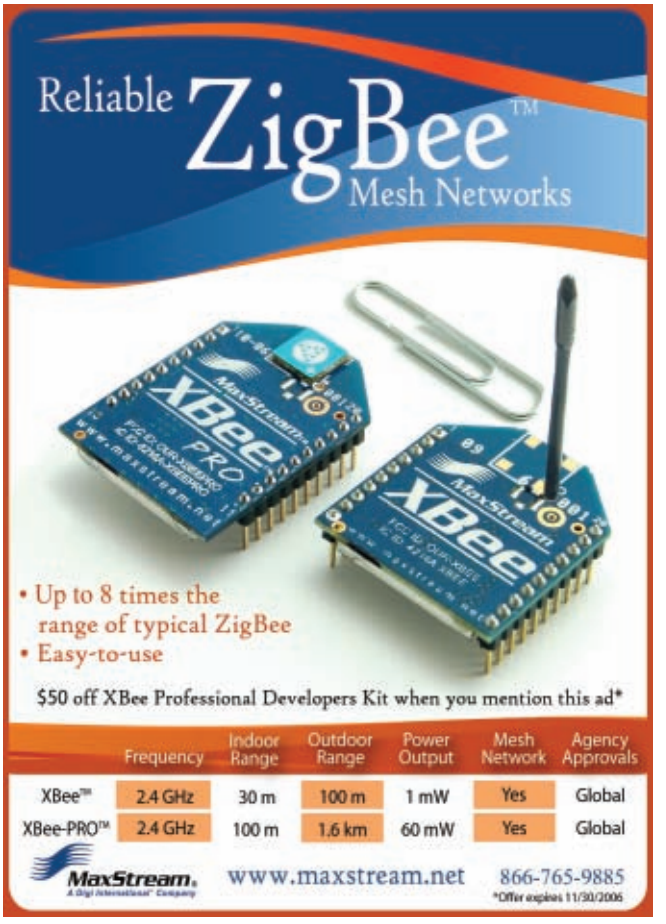
Download a free demo today, visit our website.



For a list of global offices and distributors:
www.accuratetechnologies.com
sales@accuratetechnologies.com

ATI ACCURATE TECHNOLOGIES

KVASER Advanced CAN Solutions
ATI is a distributor of Kvaser CAN interfaces



USB Flash PIC Programmer

Why buy a PIC programmer when you can customize your own? Alexander shows you how. His programmer operates on USB power and doesn't require any device drivers.

There are numerous PIC programmers on the market, most of which are predominantly based on parallel or serial port interfaces. In addition, most programmers require an external voltage source at 15 V or higher. The programmer described in this article, which I call "Eva," operates on USB power alone and doesn't require device drivers. The programmer features a Microchip Technology PIC18F2455 microcontroller, which provides hardware support for low- and full-speed USB. It also has sufficient I/O lines to perform all of the tasks required for programming (e.g., turning V_{DD} and V_{PP} on and off and driving the clock and data to the target PIC microcontroller). Photo 1 shows the complete system, which includes a USB port, a serial port, a ZIF socket, status LEDs, and more.

SYSTEM REQUIREMENTS

The Eva programmer satisfies several basic requirements. It works with a wide range of flash memory-based PICs (12F, 16F, 18F, and dsPIC) and various versions of Windows (95, 98, 2000, and XP). The same codebase is used for a desktop PC and a PocketPC, and onboard and ICSP programming is supported. It's compatible with standard RS-232 serial and USB-to-serial converters and programming times comparable to JDM and IC-Prog. Finally, note that a basic system costs less than $25 in parts, and you can use any 2.1-mm ID power adapter.

Several commercial and hobby-level programmers meet most of the aforementioned requirements, but many of them have severe problems when they're running with Windows XP. The issue is that low-cost PIC programmers rely on the direct manipulation of serial or parallel ports. The latter are disappearing. Direct access to serial port lines requires special device drivers (and they don't always work), and some serial ports have inadequate voltage/current to drive the programmer. USB-to-RS-232 converters are a possible workaround, but they have problems when directly controlling RS-232 lines. (They can also cost as much as this entire project).

The Eva is considered a prototype programmer because it performs programming and verification at a single voltage level (as opposed to a production programmer, which can vary the V_{DD} to ensure correct programming at all levels). This has been adequate for my needs and is typical of simple programmers.

I do not have a lot of experience with PIC programmers, but their overall performance seems fine. (Refer to the timing text comparisons at www.piclist.com.) In many cases (e.g., PIC16F parts), the limitation is inherent in the chip itself (specified wait times during programming), and higher performance would imply that the chip specifications are being violated. For other chips (e.g., multibank PIC18F parts), performance can be significantly enhanced by properly staging the data before each programming cycle.

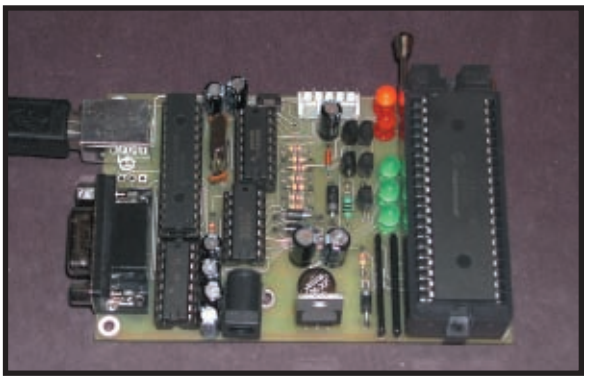


Photo 1—*Check out this programmer. It can be driven through USB or RS-232 (serial) connections and will program chips either in the ZIF socket or via a six-pin 0.1" spacing ICSP header.*

SUPPORTED PROCESSORS

A list of currently supported chips—including those that have been tested for read, write, and erase functions—is posted on the *Circuit Cellar* FTP site. The programmer has not been tested on all of the chips it supports. Chips within a particular family have the same programming algorithms, so they'll probably work if one of the family appears in the tested column.

If you want to program a chip that isn't on the list, you need to add support to the Windows software. Most of the time adding support for a new processor requires only adding it to the list of known devices and recompiling the software. For some chips, the programming algorithm doesn't match any of the supported chips. In this case, you need to write a new class, inheriting from one of the PIC16F, PIC18F, and dsPIC base classes as needed.

HARDWARE OVERVIEW

The entire Eva programmer is shown in Figure 1. A complete bill of materials, including supplier order numbers, is posted on the *Circuit Cellar* FTP site. The system costs about $30 (excluding the PCB), with the PIC18F2455 processor and the ZIF socket accounting for half the cost. Note that a fully func-

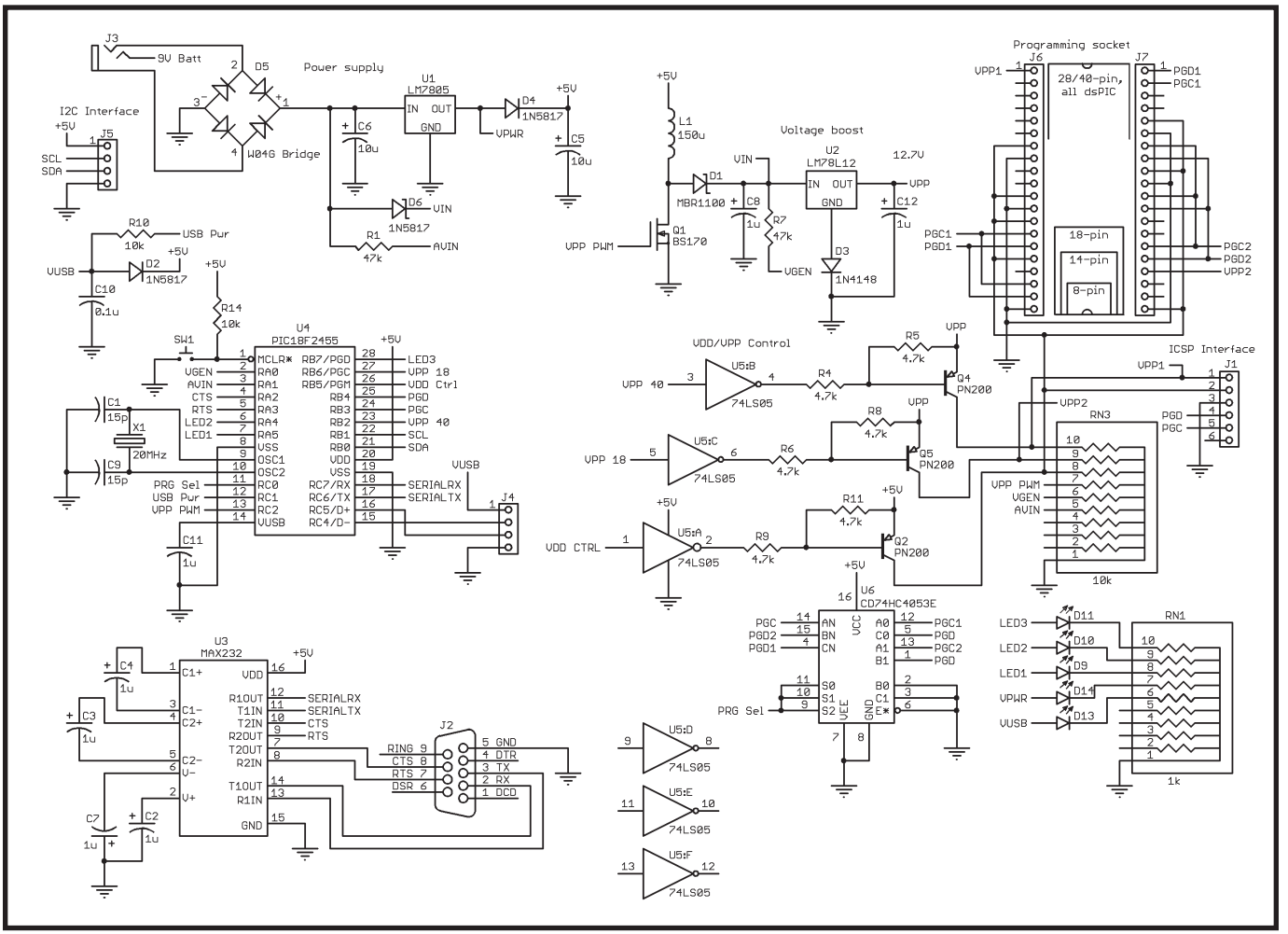


Figure 1—*The programmer is divided into functional parts: the power supply and power boost, a PIC18F2455 microcontroller with USB support, a serial connection, status LEDs, and a target chip programming interface.*

tional ICSP-based programmer can be built using only about a third of the parts (again, excluding PCB) for about $11. To build a simple USB circuit, you need only the PIC18F2455 (with crystal and capacitors), the voltage boost circuit, the V_{DD}/V_{PP} control circuit, the USB connector, the ICSP connector, and the part of the power supply hanging off the VUSB label.

With even the minimum number of parts, the programmer can be effective. Photo 2 shows the basic system programming a Microchip dsPIC30F4012. If you look at the PCB, you can see that very little is needed to build a programmer. Using only a few parts, you can build it on a small perfboard.

Programming voltage is generated with a simple DC-to-DC boost converter. Power is pulled into an inductor and then pushed into a holding capacitor through a diode. The flash memory versions of PIC processors don't require much current at the 12- to 13-V level for programming, typically less than 1 mA. This creates latitude in the inductor selected for the voltage boost. The output voltage is adjusted by varying the duty cycle of the PIC18F2455's PWM. (A longer duty cycle gives higher boost.)

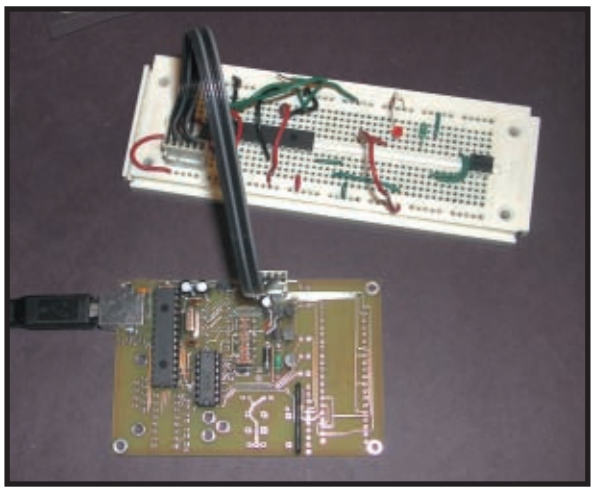


Photo 2—*You can build a simple PIC programmer with only a few parts. Here you see the programmer operating on a USB connection. It's programming a target chip through an ICSP cable. The target chip is on the prototype board and programmed in-circuit.*

The voltage boost section takes the input 4.7 V and turns it into about 12.7 V. This value is high enough to meet the minimum desired by PIC16F parts (some of which require more than 13 V), but it isn't so high that it exceeds the specification for PIC18F parts (most of which are happy at 9 to 12 V).

The total current drawn by the device is approximately 65 mA when it isn't busy, and it's as high as 75 mA during the programming of the target chip. The USB specification guaran-

tees at least 100 mA (one unit load), so the programmer is well within the boundary for a low-power device. However, you need to ensure that the target board doesn't draw excessive power when using ICSP. Your USB hub could become unhappy and shut down. Additionally, the PN200 transistor used to control V_{DD} to the target has maximum ratings of 500 mA and 625 mW, so you need to stay well under those limits in the target board to avoid damaging the programmer. If you run it on external power, you may need a heatsink on the 7805 voltage regulator.

The placement and orientation of chips in the programming (ZIF) socket are important. All dsPIC, 28-pin, and 40-pin processors are placed with their pin 1 in pin 1 of the socket (right side up and all the way at the top). All 8-, 14-, 18-, and 20-pin processors (excluding dsPIC) are placed with their pin 1 in pin 21 of the socket (upside down and all the way at the bottom).

Most of the component values in the circuit are noncritical. Some reasonable substitutions include replacing: PN200 by any small PNP transistor, 74LS05 by 74LS06, the 150-µH inductor by 100 µH, and BS170 by a PN7000 or similar N-channel MOSFET. Note that 150-, 220-, and 330-µH inductors have all been used with success (33- and 1,000-µH inductors are not recommended because they caused intermittent problems). Do not substitute a CD4053 for the 74HC4053. (The on resistance in that chip is high enough to cause problems with socket-based programming.)

Listing 1—*To customize the generic USB firmware into a specific application, the following routines are all that have to be implemented (and some may be left as stubs).*

```
int GetInputReport(byte reportID);
int SetupOutputReport(byte reportID);
int SetOutputReport(byte reportID);
int GetFeatureReport(byte reportID);
int SetupFeatureReport(byte reportID);
void SetFeatureReport(byte reportID);
```

FIRMWARE

The firmware for the PIC18F2455 is split in two parts. The first is C code that handles the USB interface, voltage boost, and serial interface. The second is assembly code that performs the actual programming duties. The reason for the split is to provide for runtime updates to the firmware. The firmware associated with sending and receiving packets doesn't change frequently. However, as additional chip support is added to the programmer, the PIC programming code often requires modification. Using this split approach, when there are changes and additions to the programming algorithms, the programmer can make changes on the fly. The structure is similar to a bootloader application.

The C code compiles under the open-source small device C compiler (SDCC). This compiler is available via download from SourceForge. Use of other compilers would require code changes. The way pointers are treated varies quite a bit between PIC C compilers. The assembly code can be com-

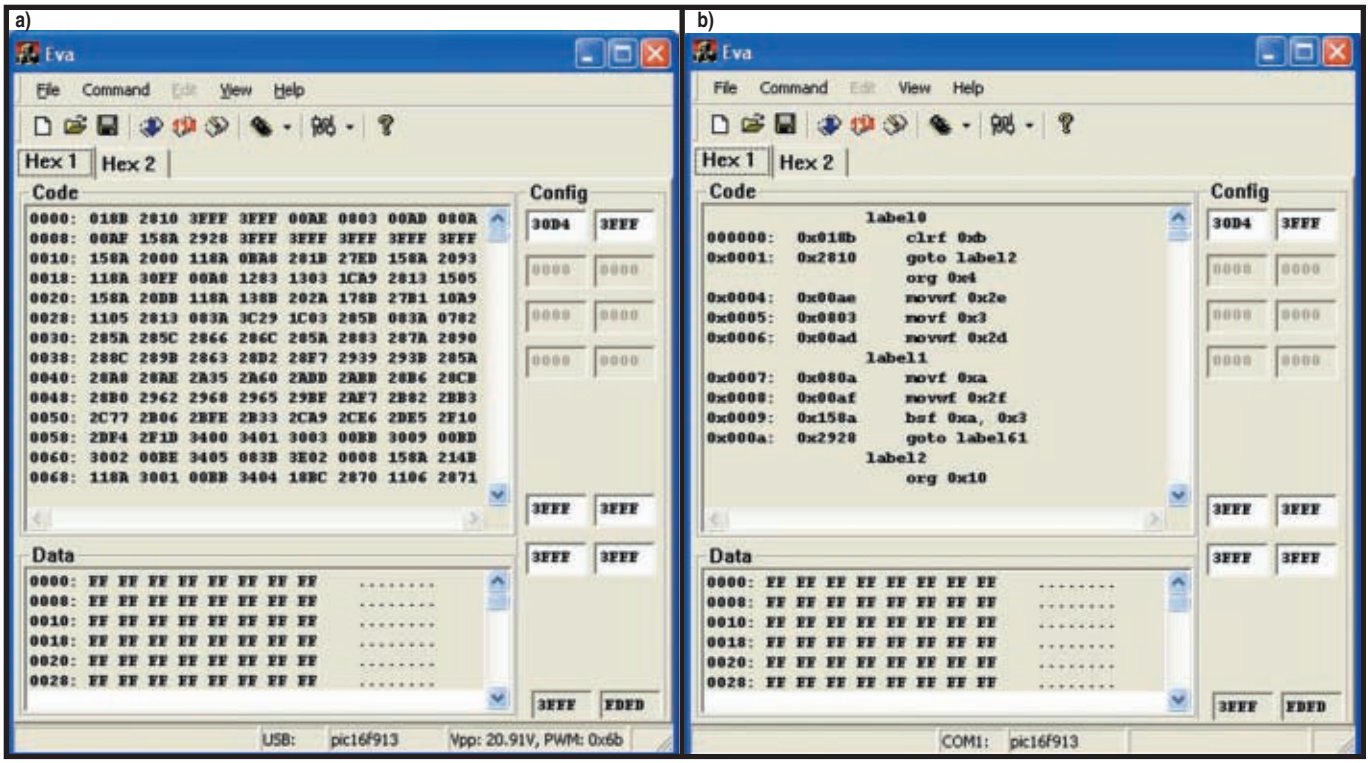


Photo 3—*The host software in Windows gives views of hex files in either raw or disassembly format for PIC16/18 families. Hex files are loaded into the software to be burned into the target. Additionally, the content of the target chip can be read into the software for viewing and validation.*

LCD DISPLAYS BY MATRIX ORBITAL









MATRIX ORBITAL produces a variety of Liquid Crystal Display modules that can accommodate a broad spectrum of applications and capabilities. Control of the modules has been simplified to a set of commands delivered to the modules via RS232 and I²C communications protocols. Matrix Orbital is one of the only display companies that offers an I²C LCD.

We have made every effort to ensure that tedious timing relationships and special conditions, typical of display controllers, are eliminated from our interface. All of our LCDs are available in numerous sizes and colors. Also with variable backlighting excellent visibility is possible in all lighting conditions. Matrix Orbital LCD displays are made to save you time and money.

SOME KEY FEATURES

- variable backlighting
- alternate temperature ranges
- 60 degree viewing angle
- long operational life
- several colors
- built in font: up to 8 user defined characters
- speeds from 1200 bps to 19.2 Kbps
- communication over serial RS-232 or TTL, I2C to USB
- 255 levels of software controlled contrast
- backlight configurable time out setting up to 180min
- general purpose outputs for a variety of applications (on most models)
- keypad interface with 12, 24 or 25 keys
- horizontal or vertical bar graphs
- extended temperature available from -20 to +70
- wide voltage power available +7V to +15V
- display specifications rated at 100,000 hrs

Get **5% off** your next purchase online at Matrix Orbital.
Simply log on to our site below and enter the code **MOCC0134** in the area provided at the check-out.

WWW.MATRIXORBITAL.COM



Photo 4—*The host software has been written so that it can be compiled as either a traditional Windows application or as a PocketPC application. As you can see, I ran my programmer on a PocketPC emulator. (Testing was performed on an HP iPAQ hx4700.) Programming is performed via a serial connection because few PocketPCs can function as a USB host.*

piled using either Microchip's MPASM or using the open-source GPUtils (also available from SourceForge).

An alternative implementation of the firmware is available for PIC16F737 and PIC16F913 processors. By selecting one of these devices in MPLAB instead of the PIC18F2455, the firmware can be built for either of them. These chips support only serial I/O, not USB. The chips also don't support runtime updates to their firmware. However, you can save a little money if you want a serial-only version of the programmer. The software is the same, regardless of the target processor.

USB COMMUNICATIONS

Using USB to communicate between a host (the PC) and a device (the programmer) is a relatively complex task. However, I wanted it to be easy for the end user to get up and running. The HID class turned out to be the best bet. In Windows, a HID doesn't need any new drivers. Nothing visible happens other than popping up a description of the device the first time it is attached (and your PC makes a bloop noise when it is attached and detached).

There are some performance issues associated with a HID versus other USB classes (e.g., bulk transfers). However, the speed is good enough (several times what you get over a 115-KB serial line) that the ease of use outweighs the performance hit. I have noticed that not all USB hubs are created equal. The overall bandwidth is several times higher on the hub in my keyboard in comparison to the one on the side of my laptop.

USB FIRMWARE FRAMEWORK

The main firmware for managing USB communications is found in the usb.h and usb.c files. This code provides a framework that closely follows the state transitions described in Chapter 9 of the USB 2.0 specification. When the device is connected to the PC's USB port, the PC and firmware go through a series of negotiations to ensure that the PC knows which kind of device is connected, the interfaces it supports, and the device/product identifiers.

The code in usb.c is not specific to the programmer. It was designed so I could reuse it in a generic manner for any USB project. It is also not quite complete by itself. You need to implement a few callback routines to manage application-specific tasks (see Listing 1, p.14). If you are not using a particular function (e.g., input reports), then you can use a stub function that does nothing. In the programmer, only feature reports are used and only those with a report ID of zero. The entry point for handling the reports is in main.c.

The callback functions are invoked at three points: the point at which the firmware realizes that a buffer is needed to store information (`SetupXXX()`) so a buffer can be allocated; when the host is requesting data (`GetXXX`); and when the host is sending information (`PutXXX()`). The programmer uses fixed buffers and responds to the `Setup` calls by setting a pointer to the beginning of those buffers.

COMMAND PACKETS

Commands are sent to the programmer and information is returned in byte packets. Every packet has the same form: <COMMAND><DATA LENGTH><DATA>. The data portion's content varies depending on the command. It may be empty for something simple like a request for the version number or quite long for a packet that is reading or writing program memory in the microcontroller. There are a total of 39 commands currently implemented in the firmware. They are roughly divided into housekeeping commands (reading the version, reading ports, and turning control lines on and off), 14-bit PIC (12F/16F series) commands (read, write, and erase), 16-bit PIC (18F series) commands, and 24-bit PIC (dsPIC series) commands.

The host software expects a strict command and response. For every command sent to the device, a packet is expected in return. If a timeout occurs while waiting for a response, it is assumed that the operation has failed. There are also circumstances when the response packet indicates a failure (e.g., a write can fail if the memory isn't erased first or the target chip is write protected).

Packets are sent through USB as feature reports on control pipe 0. After they have been read into an internal RAM buffer, the command byte is examined to see what processing should take place. This takes place in the `ProcessFeatureReport()` function in main.c. Some commands are handled right on the spot (firmware read/erase/write),

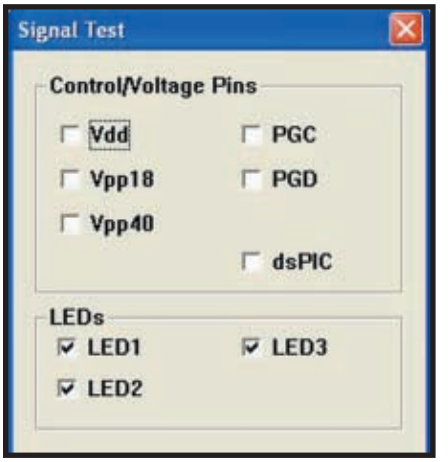


Photo 5—*The host software provides basic functionality to perform hardware checks. Each of the status LEDs can be toggled on and off, and each of the control lines used during programming can be toggled. Using this dialog and a voltmeter, you can check if the hardware is functioning properly.*

while most are vectored to the assembly portion of the firmware at byte address 0x4000 (evaprog.asm).

Adding new commands is quite simple. A new entry is added to a jump table, and the offset in the table is used as the command value. (Refer to evadefs.h for a complete listing of commands and their values.) Because that part of the firmware can be updated at runtime, the programmer can be given the new commands without difficulty.

SOFTWARE OVERVIEW

The Windows software was written in C++ with the Windows template library (WTL). It didn't require any special runtime libraries or special device drivers.

Although it is expected that most of the time you will be using a desktop or laptop PC to run the programmer, the software also supports the PocketPC. (It was tested on Windows Mobile 2003. It may also work on earlier versions.) Building the software requires Microsoft's Visual Studio.

USER INTERFACE

The host software provides the basic set of functions that you need for programming: read and save hex files; erase, read, and write target memory; select the target processor type; select the I/O port (USB or COMx); and update the firmware. The most frequently used operations are accessible via the buttons on the toolbar. Less frequently used functions (e.g., updating firmware, erasing EEPROM, and writing configuration fuses) are accessed through menu items. The host application is shown in Photo 3. On the left is a standard view of a hex file. On the right is a disassembly view of the same file. The user interface for the PocketPC is similar (see Photo 4).

When a hex file is loaded or read from the target, it is displayed in the active tab. The program memory is in the main window. The EEPROM's content (if any) is in the lower window. The configuration fuses are displayed to the right. After loading a hex file, it can be displayed as raw hexadecimal values or disassembled and displayed as PIC assembly source code. (Only the PIC12F, PIC16F, and PIC18F are applicable. The disassembly of the dsPIC is not yet supported.)

Clicking on the toolbar button that looks like an IC will perform an automated identification of the target chip. If it is a supported chip, the programmer will reconfigure itself to match the target (change the number of bits per work, the total number of words in memory, and so on). To manually select the processor, click on the arrow next to the button and a large selection of processors will pop up. When updating the firmware at runtime, you have to manually select the PIC18F2455. You can then load the new firmware and perform the update function.

DEBUG FUNCTIONALITY

In order to check the programmer's functionality, a simple debug dialog is provided in the host software. If the hardware has been correctly assembled,

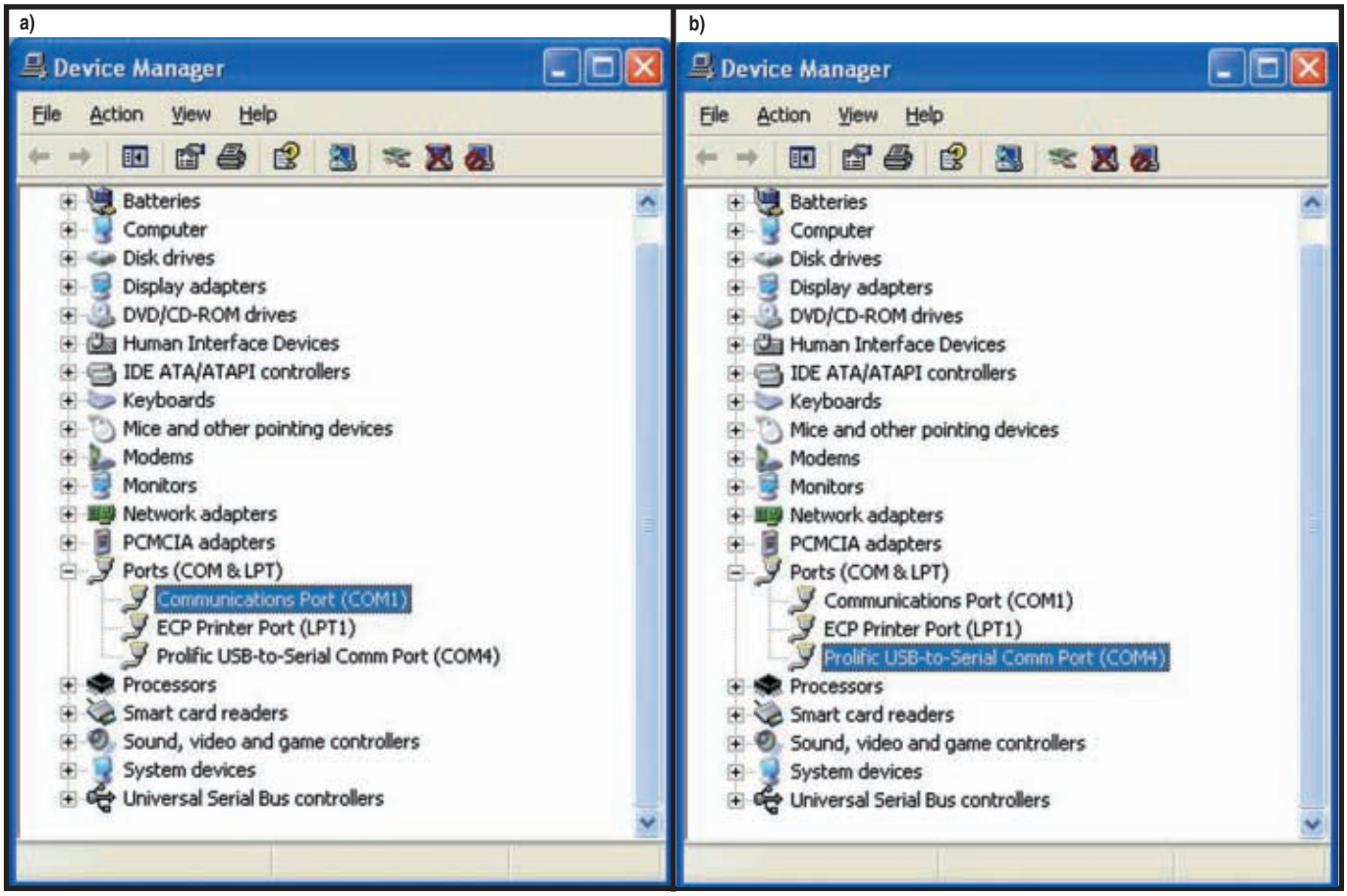


Photo 6—*To use the programmer through a serial connection, it is necessary to tell it which one to use. Using Device Manager in Windows is the easiest way to determine which COM ports are available and their names. Here you see two available ports. One is built-in (COM1:). The other is provided by a USB-to-serial converter (COM4:).*

it is possible to manually raise and lower each signal line used during programming. By checking voltages either on the ICSP header or at the corresponding pins in the ZIF socket, you can verify that everything is functioning. The dialog is shown in Photo 5 (p. 16).

Another debugging technique (used primarily when developing firmware) is to exploit the serial port to output debug information (using HyperTerminal or the equivalent). To do this, you have to recompile the basic firmware (usb.c, programmer.c, and main.c) with `DEBUG_PRINT` set to 1. (The symbol is in usb.h.) This can be a real benefit when you are trying to track down problems with USB firmware. (It's much more informative than turning an LED on or off.) Note that the initial enumeration of a USB device has some tight timing restrictions, and you don't want a lot of output going on. After that point, things are a lot less sensitive, and you can use `printf()` to expose the inner workings of the firmware.

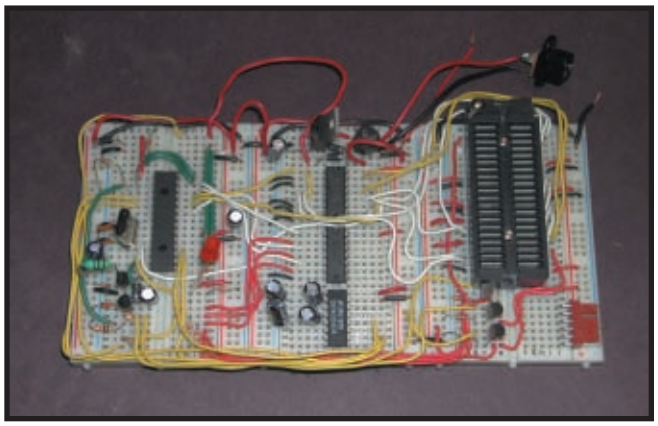


Photo 7—*A prefabricated board is not necessary in order to build the programming hardware. All of the parts are available in through-hole form and can be wired on a solderless prototyping board. (The USB and serial cables are not shown.)*

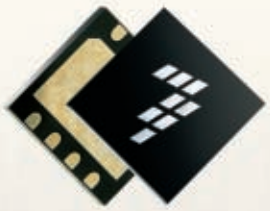
WORK THROUGH SERIAL PORTS

If you use your programmer through a serial port, you will need to know the port to which it is attached. If you are using a standard serial port on the back of your machine, it will likely be COM1:. If you are using a USB-to-serial converter, Windows will give it a port assignment. To find the port assignment, all you need to do is open Device Manager and look at the list of ports.

Photo 6 (p. 17) shows a standard serial port highlighted on the left. A port associated with a USB-Serial converter is on the right.

ADDITIONAL USES

The hardware and firmware described in this article can also be put to other uses. The hardware has LEDs, a serial port, and a header for I^2C. If you're just beginning to experiment with USB devices, you can take this as a starting point. You can build a USB-to-serial converter and use USB to control an I^2C device (e.g., an EEPROM or port expander). You can also turn lights on and off and use the voltage boost circuit output to drive op-amps. The basic firmware for USB can be used for various end applications. Simply add your code to respond to feature,



It's where 8-bit and 32-bit become compatible.

For more go to freescale.com/continuum

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. ©Freescale Semiconductor, Inc. 2006.

input, and output reports.

AVAILABILITY & LICENSE

The schematics, firmware, and Windows software described in this article are all open source (covered by the GNU GPL). You may download the code from the *Circuit Cellar* FTP site or from SourceForge on the Evausb project page. Using CVS through SourceForge is the best way to obtain the most recent software and firmware.

Every USB device is required to have a unique vendor and product ID (VID/PID). The programmer uses VID = 0x04D8 and PID = 0xFF39. This VID/PID pair was obtained through a sublicense from Microchip Technology. If you use this firmware for any purpose other than building a programmer, you must get your own VID/PID and change the device descriptor in usb.c.

The PCB I used was a prototype (only three were made). I can make it available, as well as preprogrammed PIC18F2450/2455 chips if there is sufficient interest (50 to 100 orders) to warrant the initial production cost. The board will cost you around $10. The cost of a preprogrammed PIC will be $8 (plus shipping costs). You can also implement the programmer on a breadboard (see Photo 7).

WORK IN PROGRESS

The purpose of this project was to build a USB-based PIC programmer to support a broad range of chips. Along the way, I learned a great deal about USB communications and PIC programming algorithms. I hope you'll find the open-source software and firmware a useful resource for incorporating USB in your designs.

This project is still a work in progress, so I'm open to suggestions. As I make changes, I'll add them to the CVS baseline on SourceForge.

Alexander Enzmann enjoys writing software and making things go whirr and thump. For the last few years, he's been having fun writing firmware for PIC microcontrollers and indulging in both activities. You may contact him at xan-der@comcast.net.

PROJECT FILES

To download the code and additional files, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2006/196.

RESOURCES

Cygwin, Red Hat, http://sources.redhat.com/cygwin/.

Eva on SourceForge, http://sourceforge.net/projects/evausb.

GPUtils, SourceForge, http://sourceforge.net/projects/gputils/.

PIC Information, www.piclist.com.

Small device C compiler, SourceForge, http://sdcc.sourceforge.net/.

USB Information, Lakeview Research, www.lvr.com/index.html.

USB Specification, www.usb.org.

SOURCE

PIC18F2455 Microcontroller
Microchip Technology, Inc.
www.microchip.com

Welcome to the Controller Continuum

At Freescale, we've broken the bit barrier. The Controller Continuum is our product roadmap that combines the simplicity of working with 8-bit and the high-performance of 32-bit. Soon, we'll be rolling out pin-for-pin compatible devices that let you easily upgrade your 8-bit designs to 32-bit. By sharing common peripherals and tools, such as our now easier-to-use Fast Track services for the CodeWarrior® tool suite, you can accelerate design cycles and speed products to market. It's not only how to make smart things work. It's how to work smarter.



freescale™
semiconductor

Control Panel Design

Implement a Modular Physical User Interface System

Are you tired of designing expensive custom control panels for all of your new embedded systems? Eric's user interface building block modules simplify the process.

It seems that every new piece of software coming out today features a flashier graphical user interface (GUI) than the last one. GUIs are now playing an increasingly important role in the design of embedded systems. This is made possible in large part by the availability of low-cost, high-performance SBC platforms. Many popular formats now exist, such as PC/104, EPIC, EBX, and Mini-ITX, to name only a few. Power consumption has decreased significantly with the advent of more efficient processors such as the XScale and ARM9. Ample processing power, low power consumption, and low price are all attributes that make SBCs attractive targets for embedded development.

GUI OR PUI?

Most of the new low-cost platforms provide limited means of user interface. The ultra-low-cost boards are restricted to a basic RS-232 serial port terminal. More integrated boards may feature a VGA or LCD panel output. At the higher end of the spectrum, full keyboard, mouse, and sometimes touchscreen interfaces are also available. Granted, for highly complex or general-purpose applications, the VGA monitor or graphical LCD panel-based GUI is hard to beat. However, in a lot of fixed-function embedded applications, the user-interface requirements are simple and well defined. They don't necessarily require the use of a mouse, keyboard, and monitor combination.

Personally, I am a hands-on type of person. I don't usually get too excited about dropdown menus and pop-up messages. I like a good old-fashioned physical user interface (PUI) with lots of knobs, switches, and flashing LEDs! After being frustrated with the high cost of designing custom control panels for each project, I had an idea. Why not design a series of generic low-cost modular user interface building blocks? These building block modules would be arbitrarily interconnected to form a more complex user panel. Each module would feature some basic user interface devices, such as mechanical rotary encoders, potentiometers, selector switches, toggle switches, push buttons, LEDs, an LED display, and an LCD. The various modules could be strung together in a daisy-chain fashion using a simple protocol based on the popular RS-232 serial interface that's available on all embedded platforms, including 8-bit development boards.

The advantage of the modular approach is that the same set of boards can now be reused and reconfigured without incurring NRE charges. After giving it much thought, the IOSTRING system was born!

STRING 'EM TOGETHER

The IOSTRING system consists of a series of unique PC board modules that can be daisy chained into arbitrary configurations (see Figure 1). The system currently comprises three different board configurations. Two input type boards are called IOS-IP-01 and IOS-IP-02. One output type of board is called IOS-OP-01. Any of these boards can be strung together in a daisy chain configuration. Multiple boards of each type can also be used in the string. The serial port of the host computer is connected to the first module, whose output is connected to the input of the next module, and so on. The last module in the chain must have a loopback jumper installed to send its output back to the host computer, thereby closing the chain.

Each of the modules is controlled by an on-board, 8-bit microcontroller. This microcontroller is responsible for monitoring the user I/O devices, repeating RS-232 messages around the

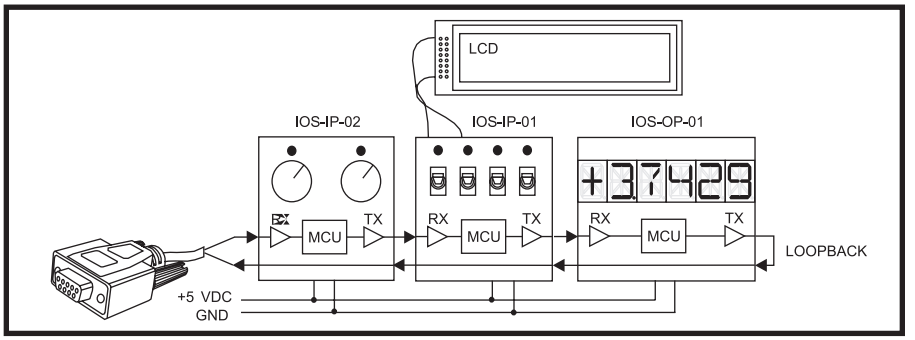


Figure 1—*Each IOSTRING module contains a certain number of basic user interface devices, such as push buttons, rotary encoders, toggle switches, selector switches, potentiometers, LEDs, and displays. Each module can be daisy chained to the next module to build a full-fledged user interface. The IOSTRING protocol, which is based on RS-232, ensures that the host computer can uniquely address all of the modules.*

string, and responding to messages that are addressed to it. When the host sends a message to the first node in the string, this node first reads the message and checks whether the message is addressed to it by comparing the DEST_ADDR field with the HOP_CNT field. If not, it forwards the message packet to the next node after having increased its HOP_CNT field. This process repeats itself until the last node in the string, which then sends the message back to the host processor through its installed loop-back jumper.

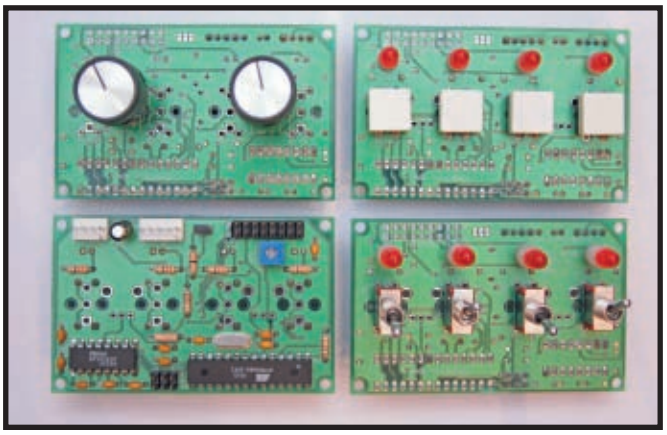


Photo 1—*The IOS-IP-01 board has numerous configurations. Starting at the top left and moving clockwise, you can see two potentiometers, four push buttons plus four LEDs, or four toggle switches plus four LEDs. You can also see the back of the board. These various button configurations can also be mixed and matched by soldering different button types on each board.*

Inherently, this technique makes the nodes in the chain self-enumerating. Individual node addresses don't need to be configured with dip-switches because the addresses are automatically deduced based on the node's position in the string. At power-up, the host can either interrogate each node to find out which type of board is located in each position, or it can simply assume a hard-coded node configuration.

CLOSE-UP EXAMINATION

Let's now take a detailed look at the circuitry. The IOS-IP-01 can accommodate up to four push buttons, four toggle switches, and two potentiometers (see Figure 2). It can also provide up to four status LEDs as well as interfacing to an off-the-shelf character LCD module. The board supports most standard character LCDs on the market. The display's backlight can be switched on or off through software. This is handy for saving power or blinking the display. An external regulated 5-VDC supply powers the board. A low-cost Atmel ATmega88 microcontroller is used at the heart of each board. The RS-232 data rate is set at 19.2 kbps.

In order to maintain a compact board size and keep the switches centered, not all of the switches and potentiometers can be installed simultaneously on the board. Dual footprints accommodate either a push button or a toggle switch in each position, but not both at the same time. As an alternative to the switches or push buttons, any dry contacts can be accommodated. This enables you to use even more user interface control panel options if you want.

A locking Molex connector makes the RS-232 connection to the board. The RX input connector has four positions. The TX output connector has five positions. This ensures that the wrong end of the cable cannot be plugged in by mistake, which could short the outputs together. Photo 1 shows the IOS-IP-01 board and some of the many possible configurations.

The circuitry for the IOS-IP-02 board is shown in Figure 3 (p. 22). This board can accommodate up to two rotary shaft encoders, each with integral push buttons. It can also read up to two, 12-position rotary selector switches. Finally, it can accommodate two LEDs. For the same reasons as I mentioned before, only two devices of either type can be loaded at one time.

A picture of two possible configurations of the IOS-IP-02 board is posted on the *Circuit Cellar* FTP site. The top of the image shows an IOS-IP-01 board connected to an off-the-shelf Optrex 2 × 20 character LCD. The bot-

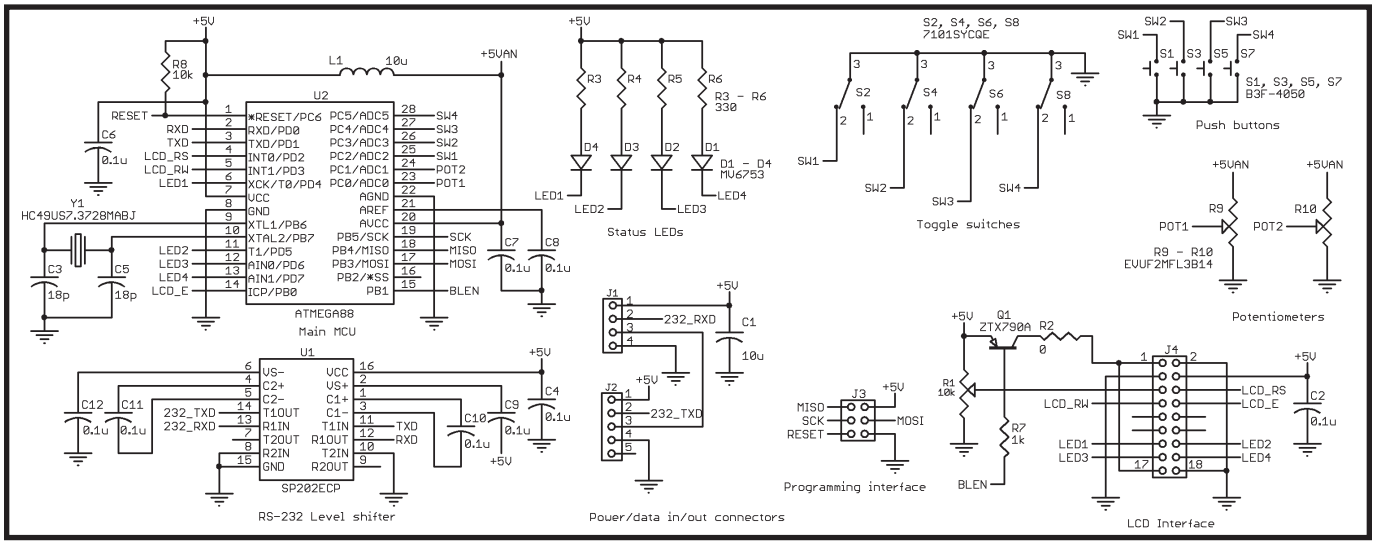


Figure 2—*The IOS-IP-01 input type of board can read up to four push buttons, four toggle switches, and two potentiometers. It can provide up to four status LEDs as well as a full connection to an LCD with backlight control. To keep the board form factor small and the buttons and switches centered, not all devices can be installed simultaneously.*

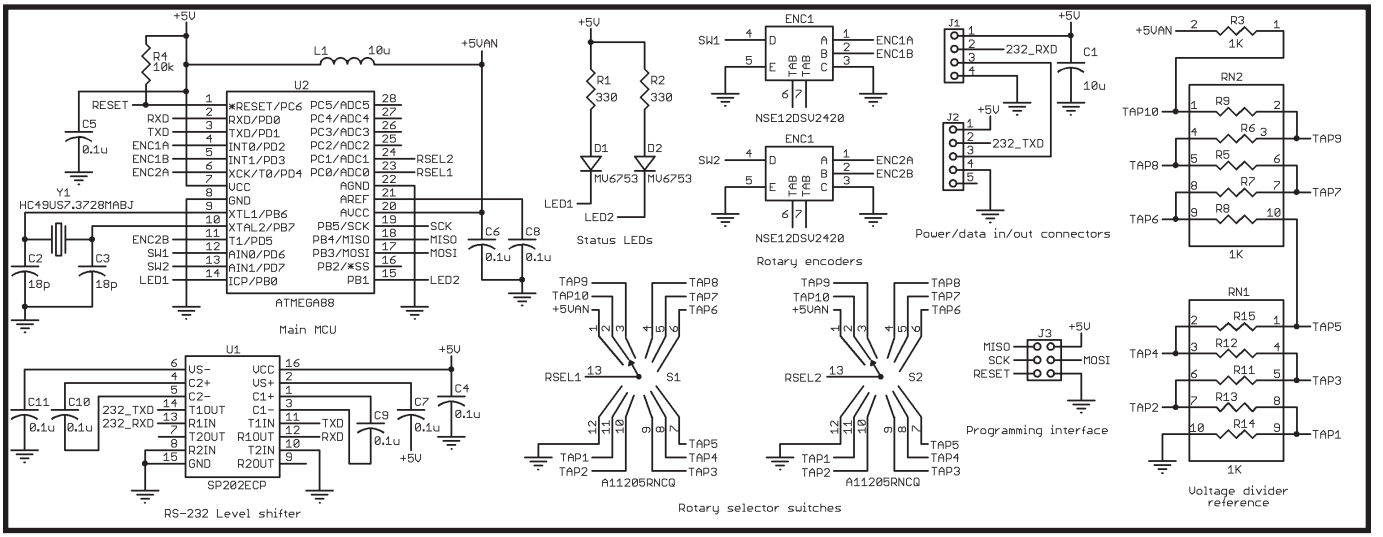


Figure 3—*The IOS-IP-02 input type of board can read up to two rotary shaft encoders, each with integrated push buttons (activated by pressing down on the shaft). It can also read up to two rotary selector switches and provide up to two status LEDs. Only two input devices can be installed at one time.*

tom image is of two of the many possible configurations of the IOS-IP-02 module. A pair of 12 position rotary selector switches is on the bottom. On the bottom right is a pair of rotary encoders with integrated push buttons.

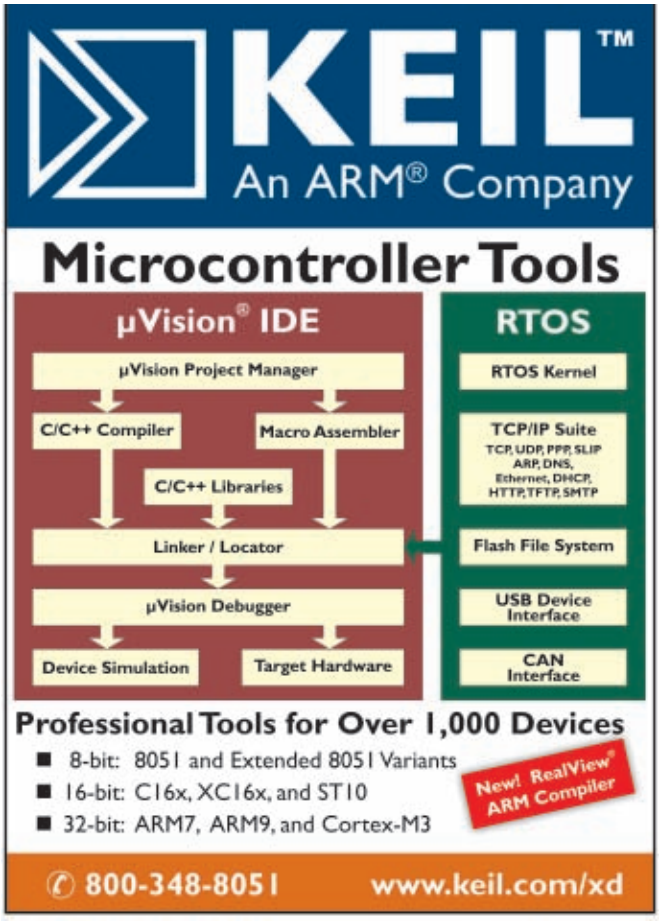
Figure 4 (p. 24) shows the details of the IOS-OP-01 board. The board is a scanned six-digit alphanumeric LED display that features an Allegro A6276 current-regulated LED driver for the display segments. This ensures an eye-catching consistent intensity profile across all of the characters. A potentiometer enables you to adjust the display's brightness.

A picture of the scanned six-digit LED display board is available on the *Circuit Cellar* FTP site. The board was designed so that the LED digits are flush with the edge of the board. This makes it possible to mount a number of these units side by side to increase the message display size.

PROTOCOL PLEASE

At the heart of the IOSTRING sys-





RabbitFLEX

A New Way To Customize





- **Click-to-ship in 5 days!**
- **Pay only for what you need**
- **Revision friendly**
- **Perfect for prototype and production**

FLEX Ready

RabbitFLEX™ is an unique build system that gives you the power to develop custom boards without the hassle and the cost. The RabbitFLEX simple-to-use web interface allows you to choose from numerous options such as digital I/O, analog I/O, serial ports, and Ethernet connections on your custom board. Just configure and buy online and our patent pending manufacturing process will deliver your solution in a matter of days. With RabbitFLEX you will reduce design risk, manufacturing cost, and development time.

Start developing now by ordering the RabbitFLEX Tool Kit and your own custom RabbitFLEX board. Take your solution to the next level.

Configure and Buy Online
www.rabbitFLEX.com

Quick-Turn Boards Range From **$149-$279**

RabbitFLEX Tool Kit **$199**

Test Drive RabbitFLEX

Build your custom RabbitFLEX board online. Add a tool kit to your order for a complete development system including Dynamic C®.

www.rabbitFLEX.com



For a limited time with kit purchase.

RABBIT Semiconductor

2900 Spafford Street, Davis, CA 95616 Tel **530.757.8400**

Solutions That Work

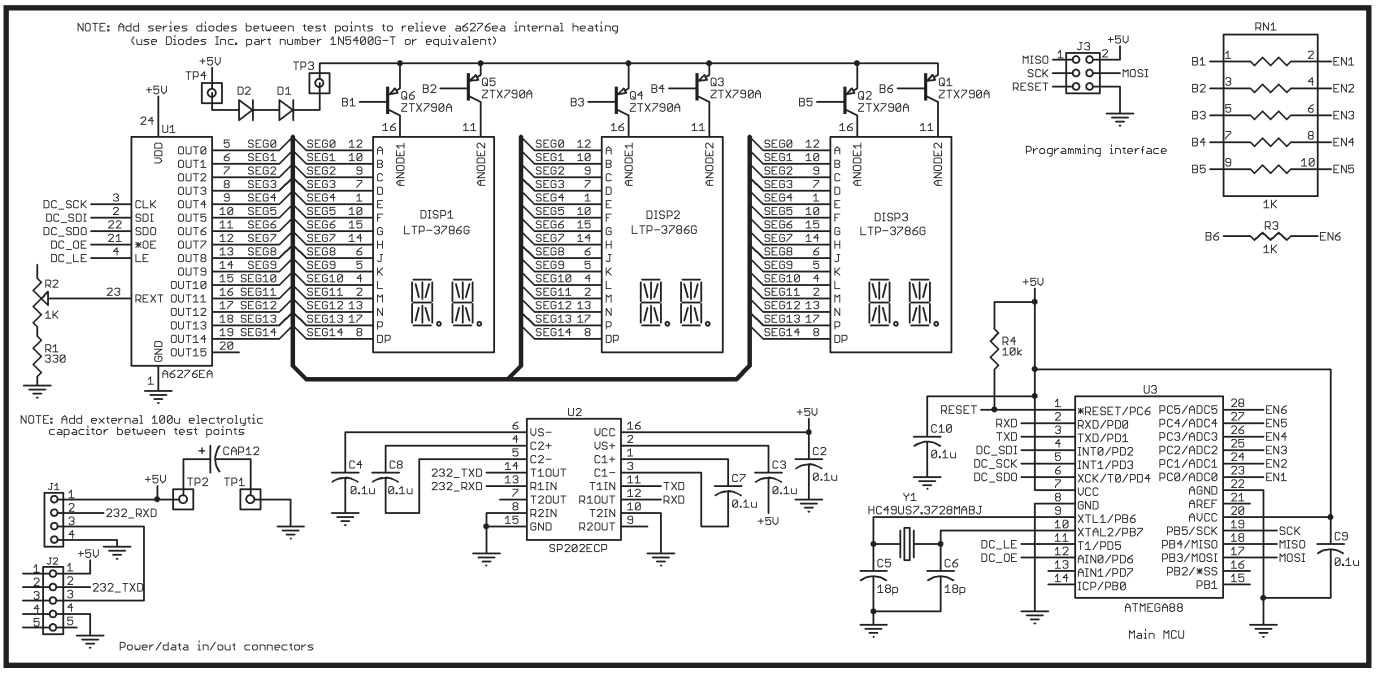


Figure 4—*The IOS-OP-01 output type of board is a scanned alphanumeric LED display with six digits. The display's brightness can be adjusted with a potentiometer. Several displays can be mounted side by side to extend the message size.*

tem is a streamlined, robust serial protocol based on RS-232. Although it would have been possible to use alternate multi-drop networking options such as RS-485 or CAN, very few embedded computers offer native support for those other types of interfaces. The simplest choice seemed to be using RS-232. Of course, the final protocol needed to be able to control more than one device to be scaleable, generate interrupt-type messages, detect failures, and be free of hardware address setup schemes. In an attempt to accommodate all of these requirements, I came up



SERIAL COMMUNICATIONS SPECIALISTS

Test and Measurement Solutions

EasySYNC™

featured products

CAN-USB
USB - CAN Bus adapter
$154



CAN-232
RS232 - CAN Bus adapter
$108

Affordable CAN Bus Solutions from $108 (CAN-232)

CAN-USB and CAN-232 are small adapters that plug into any PC USB / RS232 Port respectively to give instant CAN connectivity. These can be treated by software as a standard Windows COM Port. Sending and receiving can be done in standard ASCII format. These are high performance products costing much less than competitive solutions.

=== Bronze Prize Winner ===
NASA Tech Briefs 2004
Products of the Year

ANT16
16 channel logic analyzer
- probe set extra
$220



DS1M12
2 channel 1MS/s PC scope, signal generator & data logger
$333

USB Instruments - PC Oscilloscopes & Logic Analyzers

Our PC Instruments may be budget priced but have a wealth of features normally only found in more expensive instrumentation. Our oscilloscopes have sophisticated digital triggering including delayed timebase and come with application software and DLL interface to 3rd Party apps. The compact ANT8 and ANT16 Logic Analyzers feature 8/16 capture channels of data at a blazing 500MS/S sample rate.

Low Profile Version also available



ES-D-1004
4 Port UPCI RS232 Serial Card
Spider Cable or COMBOX IO (extra)
$99.00

ES-4C-LP
Spider Cable accessory for ES-D-1004
$11.95

uPCI Serial Cards from $19.45 (ES-D-1001)

Discover our great value for money range of quality multi-port RS232, RS422 & RS485 uPCI serial cards and accessories including opto-isolated versions. All feature a wide range of O.S. support.



ES-U-1002-M $59.45
2 Port Industrial USB RS232 Serial with wall mount bracket and 5V DC auxiliary output

**** NEW PREMIUM PRODUCT ****
$22.50



ES-U-1001-R10 GOLD
ES-U-1001-R is our new premium USB to RS232 adapter cable. Its high gloss white enclosure and side-lit blue traffic indicators are designed to appeal to discerning PC and MAC users alike.

1 to 16 port USB to Serial Adapters from $18.48

With over 20 different models available, we probably stock the widest range of USB Serial Adapters available anywhere. We offer converter cables, multi-port enclosure style models in metal and plastic, also rack mount units with integral PSU such as the ES-U-1016-RM. Serial interfaces supported include RS232, RS422 and RS485. We also supply opto-isolated RS422 and RS485 versions for reliable long distance communications. All our USB Serial products are based on the premium chipsets and drivers from FTDI Chip for superior compatibility, performance and technical support across Windows, MAC-OS, CE and Linux platforms.

ES-E-3008-M
$559



8 Port Industrial Ethernet RS232 / RS422 / RS485 Serial Server with wall mount bracket and PSU.

ES-W-3001-M
only $199
***** NEW *****



Single Port high performance Industrial Wireless Ethernet RS232 / RS422 / RS485 Serial Server with PSU and wall mount bracket. Connects wired also.

EtherNet & Wi-Fi 802-11b/g RS232/422/485 Serial Servers

One to eight port industrial strength Ethernet and Wireless ethernet serial RS232/RS422/RS485 Servers. Connect to your serial device remotely over your Wireless network, Ethernet or via the Internet. Based on 32-bit ARM™ CPU architecture, these systems offer powerful serial connectivity and a wealth of features. WLAN models comply with IEEE 802.11b/g, max. 54 Mb/s and also offer a 10/100Mbps secondary ethernet connection. All models come complete with PSU.
Prices start at only $119.95 (ES-E-1001-M).

EasySync Ltd

5285 NE Elam Young Parkway, Suite B800
Hillsboro, OR 97124-6499 USA
Tel: (503) 547-0909 Fax: (503) 547-0990
E-Mail : sales@easysync-ltd.com
Web : www.easysync-ltd.com

with the IOSTRING protocol.

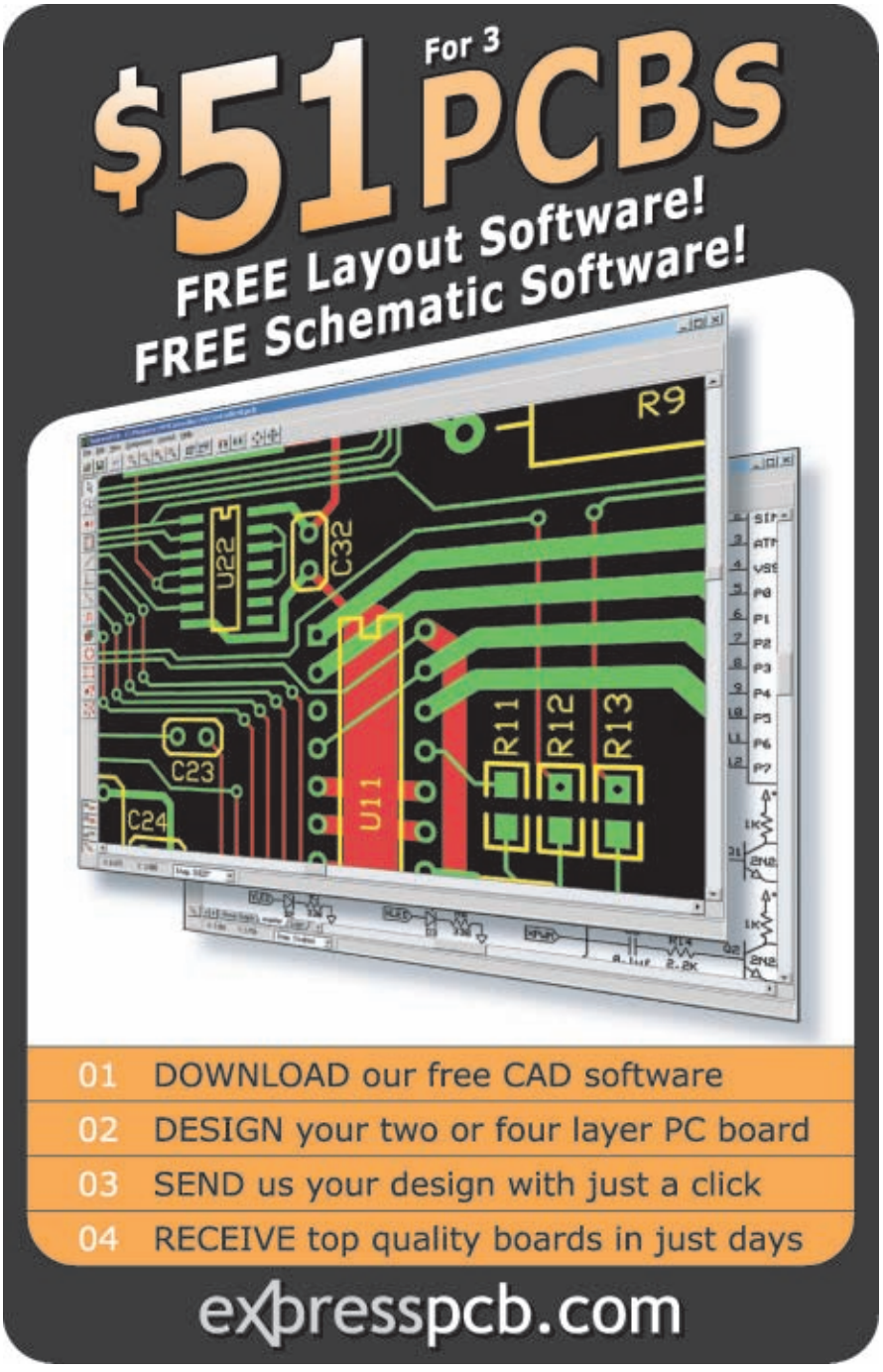
Figure 5 (p. 28) shows the basic binary packet structure of a message. Each node in the string, including the host computer, can generate a message at any time and forward it to the next node in the string. Subsequent nodes then repeat the message until it reaches its ultimate destination address. Adequate internal buffering is provided inside each node in case it receives a message while repeating a previous one. The ability of each node to spontaneously generate a message means that the system is not polled. Instead it can generate "interrupt-type" messages only when events are detected (e.g., when you press a push button). This frees the host system from having to perform time-consuming polling operations.

The first byte in the packet is the fixed header (HDR) value 0xFF. The next byte is the header type (HDR_TYPE) field, which describes the type of message. This field takes a value between 0xF0 and 0xFF. Any occurrence of 0xFF following the HDR_TYPE field must be followed by an appended 0x00 (byte stuffing) by the sending party to ensure the uniqueness of the "0xFF header + nonzero type" marker in the event of message corruption.

On the receiving side, any occurrence of 0xFF followed by 0x00 will see the 0x00 stripped from the message before the message is processed (except of course if this 0xFF legitimately belongs to the HDR, in which case it will be immediately followed by a value between 0xF0 and 0xFF corresponding to the HDR_TYPE field). Generally speaking, the message sizes will be nominally 16 bytes with potentially some extra 0x00 values added between the bytes depending on the occurrence of 0xFF.

Four message types are currently supported in the protocol. When the master sends a broadcast message, each node responds to it and also forwards the message to the next neighboring node. Here the term "master" refers to the host computer. The host computer uses a master-to-slave message to communicate with a specific node. A slave reply to the master message is returned by a slave to the master in response to a query by the master. Any nodes in between simply repeat the reply message until it makes its way around the chain back to the master. Finally, a slave generates a slave-initiated message. It's usually destined to the master each time a node detects some activity on its user interface devices (e.g., a user button being pressed or the turn of a potentiometer).

The next field in the packet is the hop count (HOP_CNT). As the name implies, it starts as a zero value when the message leaves the master (or when it leaves a slave node that initiates a message). Each node that subsequently receives the message must preincrement this field before comparing to the desired destination address (DEST_ADDR) field (and of course recalculating the CSUM). If the values match, then the node in question is the intended recipient of the message. It then processes the message and "swal-



lows it" by not repeating it. Of course, if the master computer requests a reply from the node, the node may thereafter send an independent slave reply to master message where applicable.

No matter what, the HOP_CNT field is always incremented through each node that it traverses. This idea is very versatile. For example, if the host needs to find out how many nodes are in the chain, it simply sends a PING message addressed to zero (i.e., itself). The message will traverse the entire chain. The HOP_CNT will increment as many times as there are nodes in the string. This can be done at power-up to detect the total node count. An alternate scenario exists when a node generates a slave-initiated message to the master. This node also sets the HOP_CNT field to zero because it initiates the message. By the time the message gets around to the master, it contains the number of nodes it traversed on the way. Essentially, this corresponds to the backwards hop count from the master back to the slave node. To then calculate the proper forward-looking node address, you simply subtract that value from *n*, which represents the total number of nodes in the string.

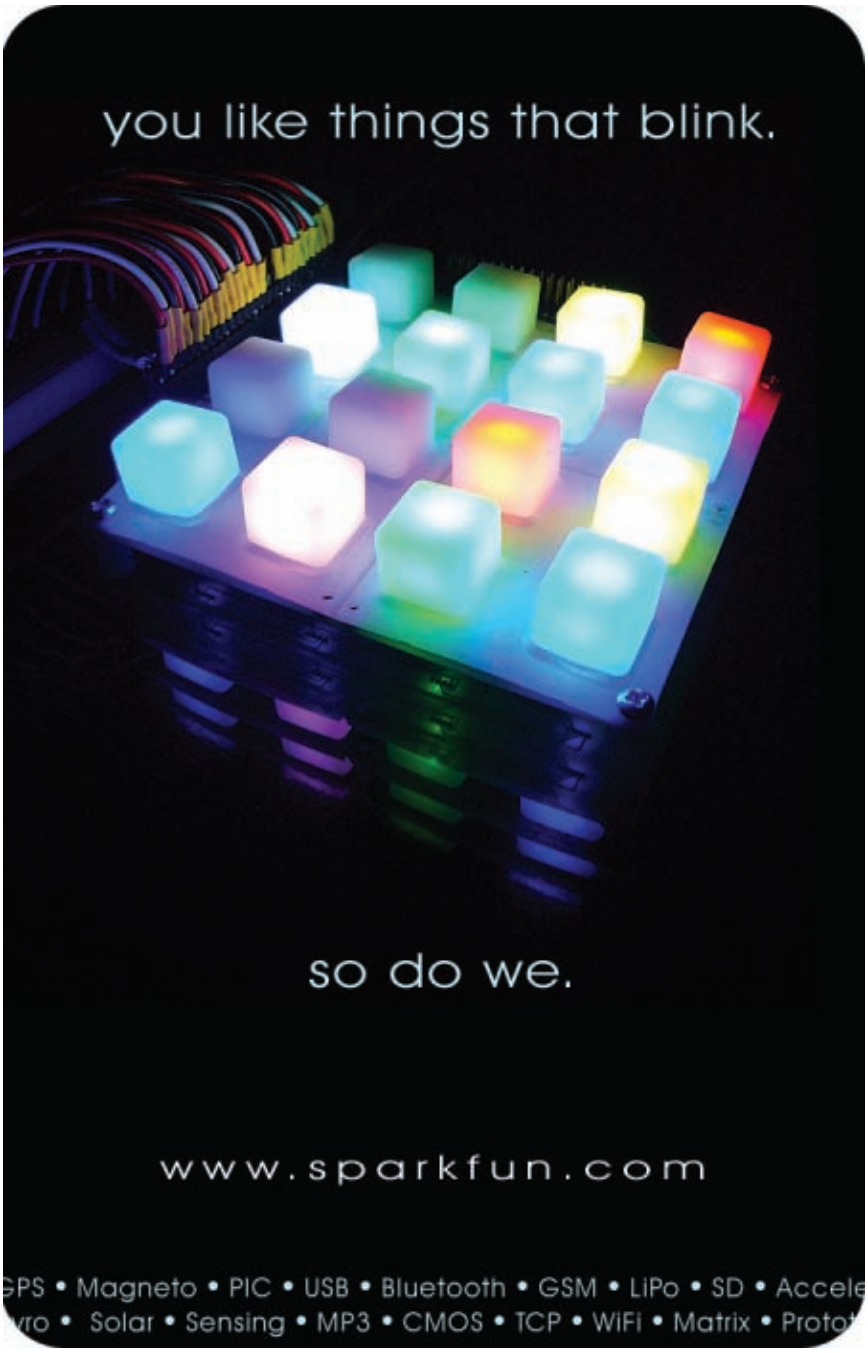
The next two data bytes are the command name (CMD_MSB/CMD_LSB). There is a long list of supported commands. Some commands are common to all boards. CMD_PING returns board type information. CMD_RESET forces a board reset. CMD_READ/WRITE_EEPROM reads and writes to the on-board user EEPROM area. Each board has an internal nonvolatile EEPROM memory to hold arbitrary user-defined parameters such as textual descriptions of the board function.

Many specific commands are provided for each of the different board types. An example is CMD_IP01_POT_EVENT, which is sent automatically by the node each time the potentiometer moves by more than a certain deadband amount. CMD_IP02_SET_LED enables you to set the state of the LEDs on that same board. LED adjustment parameters enable you to set a blinking interval and the number of blinks. Alternatively, you can simply set each LED on or off. Finally, a checksum (CSUM) parameter is located at the end of the message to ensure a reliable protocol.

HOST DRIVERS

In order to support the IOSTRING protocol on the host side, I developed a full platform-independent library in ANSI C called IOSLIB. The full library source code is posted on the *Circuit Cellar* FTP site. The library can be ported to different platforms with minimal work. The main customization required is to initialize the serial port and implement the `ioslib_send_byte()` function call to send characters out the serial port. The `ioslib_receive_bytes()` function must be called when characters are received from the serial port.

Using the IOSLIB library is straightforward. Every board connected in the string will generate messages in response to user inputs. Specific messages received by the master will invoke associated user-defined callback functions. In other words, when you press a button, a simple function



call will be automatically invoked inside the host. You are free to perform any processing needed inside this function call. You have full access to the node structure variable, which holds current node state information. Listing 1 shows a minimal implementation on a typical host system using IOSLIB.

In the example source code, a minimal system configuration consists of a single node of type IOS-IP-01 at address 1 tied directly to the master host computer. A single structure variable is allocated to hold all of the state and data parameters, as well as any registered function callback pointers associated with this node. The node can generate a large number of different messages. Each of these unique messages will call a corresponding function callback when received by the master, assuming one has been registered. Otherwise, the message is ignored.

To keep the example simple, only two callback functions were created. The first one is called when push button 1 is pressed. The other is called when it's released. The two callback functions must be registered with the system before they can be called. Note that it's also possible to register the same function callback for any number of messages.

Walking through the `main()` routine, the IOSLIB libraries are first initialized. Each of the created node structures (a single one in this example) are also initialized before they can be used. After this, both function callbacks are registered so that they become associated with the "button 1 pressed" and "button 1 released" messages received from the node. Finally, the node structure itself is registered with the system. This simply adds it to a list of nodes in the system and records its node address before the IOSLIB system libraries can take over to automatically handle the incoming and outgoing messages. After the system initialization is complete, the system begins its main loop of execution. The only requirement beyond that point is to periodically call the `ioslib_protocol_process_msg()` function to service the IOSLIB message queue.

To send a master-to-slave message, a simple function call is made from a list of many supported functions by providing a pointer to the structure variable corresponding to the node to be addressed. The node's physical address is automatically extracted by the IOSLIB libraries from the structure variable. More detailed information about all of the supported functions can be found in the IOSLIB library documentation.

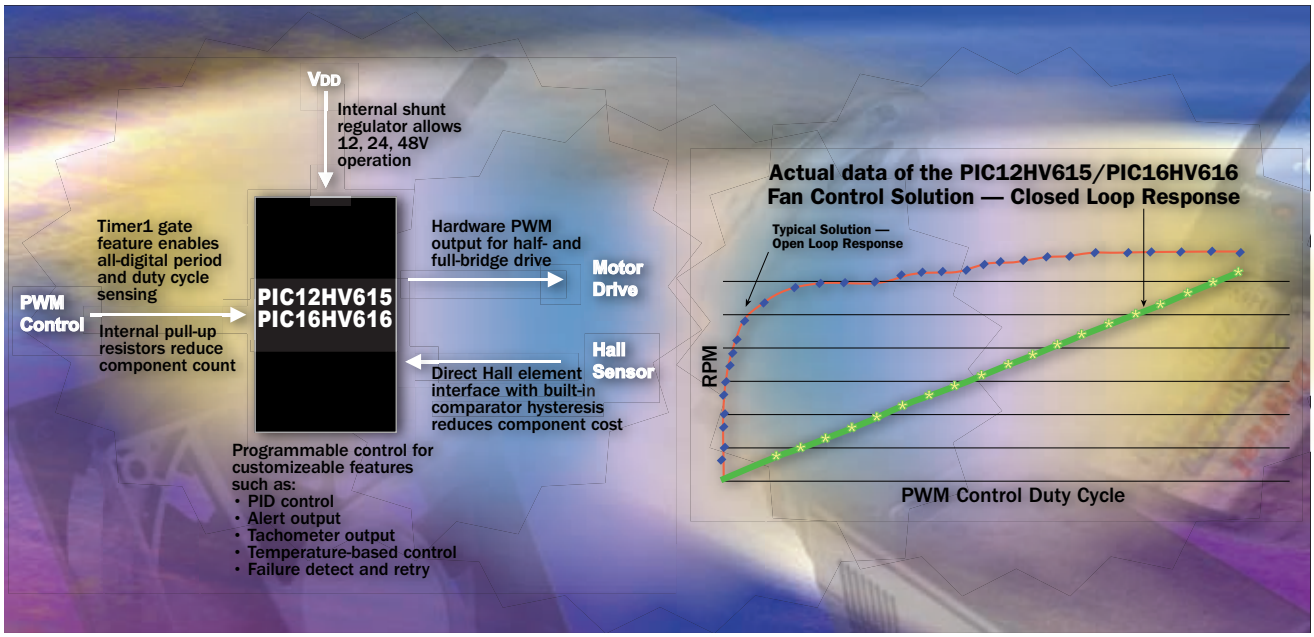
SAMPLE PROJECT

It's time to get your hands dirty!

0	1	2	3	4	5	6	7	8
HDR	**HDR_TYPE**	**HOP_CNT**	**DEST_ADDR**	**CMD_MSB**	**CMD_LSB**	**MSG_DATA_0**	**MSG_DATA_1**	**MSG_DATA_2**
	9	10	11	12	13	14	15	16
	MSG_DATA_3	**MSG_DATA_4**	**MSG_DATA_5**	**MSG_DATA_6**	**MSG_DATA_7**	**MSG_DATA_8**	**MSG_DATA_9**	**CSUM**
IMPORTANT: After the HEADER_TYPE field, any occurrence of 0xFF must be followed with an appended 0x00 to ensure the uniqueness of the 0xFF header marker. (These added values have no impact on the checksum.)								
(The added 0x00 values will be stripped off automatically during the decode process using the reverse logic.)								
HDR	HEADER	**[0xFF]**		**HOP_CNT**	HOP_COUNT	**[0-255]**		
0xFF	Fixed header value			Incremented by 1 on each hop through a slave (before being processed internally by each slave)				
HDR_TYPE	HEADER_TYPE	**[0xF0-0xFF]**		**DEST_ADDR**	DEST_ADDR	**[0-255]**		
0xFF	Broadcast Message (Master to all)			Slave destination address (slave whose HOP_COUNT matches the DEST_ADDR will be the recipient of the message)				
0xFE	Master to Slave Message			Special case when DEST_ADDR=0x00 always addresses the Master				
0xFD	Slave Reply to Master Message (only one should be in transit at any given time)							
0xFC	Slave Initiated Message to Master			**CMD_MSB**	CMD_MSB	**[0-255]**		
0xFB	<Reserved 11>							
0xFA	<Reserved 10>			High byte of the command				
0xF9	<Reserved 9>							
0xF8	<Reserved 8>			**CMD_LSB**	CMD_LSB	**[0-255]**		
0xF7	<Reserved 7>							
0xF6	<Reserved 6>			Low byte of the command				
0xF5	<Reserved 5>							
0xF4	<Reserved 4>			**MSG_DATA_0-9**	MSG_DATA_0-9	**[0-255]**		
0xF3	<Reserved 3>							
0xF2	<Reserved 2>			10 bytes of message-specific data values (set to 0x00 if not needed)				
0xF1	<Reserved 1>							
0xF0	<Reserved 0>			**CSUM**	CHECKSUM	**[0-255]**		
				Modulo 256 checksum of all the terms in the message from positions 1 to 16 (not including 0xFF header)				

Figure 5—*Take a look at the various packet fields and value ranges. Any data values equal to 0xFF following the HDR_TYPE field must be stuffed with a 0x00 immediately after. This ensures the uniqueness of the 0xFF header for quick resynchronization in the event of data corruption. Any recipient of the message is responsible for unstuffing the message by removing any occurrence of 0x00 following 0xFF prior to processing the message.*

PIC12HV615/PIC16HV616 Closed Loop Fan Speed Control



Advantages of the PIC® Microcontroller Fan Control Solution:

- Meets or exceeds the latest industry specifications
- Provides closed loop linear control
- Software-programmable dynamic response
- High level of integration provides simplicity and cost savings
- Offers high voltage support with integrated shunt regulator
- User-enabled comparator for hysteresis direct interface with a Hall Element

Now available for purchase on....

microchip DIRECT

www.microchipdirect.com

Get started with the MPLAB® In-Circuit Debugger (ICD) 2 Module for only $159.99 USD.



	High Voltage Support	Program Memory	10-bit ADC (channels)	Comparators	Enhanced CCP*	Internal Oscillator
PIC12F615	—	1.75 KB/1 Kw	4	1	Half bridge	8 MHz
PIC12HV615	Yes	1.75 KB/1 Kw	4	1	Half bridge	8 MHz
PIC16F616	—	3.5 KB/2 Kw	8	2	Full bridge	8 MHz
PIC16HV616	Yes	3.5 KB/2 Kw	8	2	Full bridge	8 MHz

* Capture, Compare, PWM

For fan control application notes, reference designs and related code examples, visit ***www.microchip.com/fancontrol***. Microchip also offers a full line of stand-alone fan controllers and fan fault detectors.

Now Pb-free! RoHS Compliant

MICROCHIP

www.microchip.com/fancontrol

Microcontrollers • Digital Signal Controllers • Analog • Serial EEPROMs

The Microchip name and logo, the Microchip logo, MPLAB and PIC are registered trademarks of Microchip Technology Incorporated in the USA and in other countries. © 2006 Microchip Technology Incorporated. All rights reserved.

Photo 2 shows the example development system. It is based on an off-the-shelf Atmel STK500 development board serving as the host. An ATmega128L microcontroller is installed in the development board by using the optional STK501 top module. Of course, any embedded computer platform with a serial port will do the job equally well, but this 8-bit host platform serves to demonstrate the minimal system resource requirements for handling the IOSTRING protocol. I ported the IOSLIB library to the STK500 platform and compiled with the ImageCraft ICCAVR C compiler. Three nodes were used in the network: one IOS-IP-01 with an attached Optrex character LCD module, an IOS-IP-02, and finally an IOS-OP-01.

For the demonstration, I instructed the STK500 host to send a "Hello World" message to the LCD after detecting a button press on the shaft encoder knob. As each of the push buttons of node 1 is pressed, the host callback function immediately sends a message to blink the LED above the respective button. Finally, as one of the shaft encoders is rotated, the callback writes a numeric character in all six positions of the LED display. The value of the digit increases at each increment of the shaft encoder.

A final step not shown here would be to mount the various IOSTRING boards into a machined or laser-cut metal or plastic front panel with an engraved legend. There are various relatively inexpensive services available on the 'Net for building custom front panels. Some companies like Front Panel Express even offer free CAD design software.

IOSTRING APPLICATIONS

What first started as a desire to lower the prototyping cost and increase the flexibility of custom control panel designs ultimately evolved into the creation of the IOSTRING system. With it, basic user interface module building blocks can be interconnected to form more intricate user control panels. The economies realized by using this approach are especially noticeable during prototyping and small volume production because NRE prototyping and the SMT assembly charges of a custom panel typically cost more than $1,000. By contrast, the individual IOSTRING modules can be built for less then $30 each depending on the configuration.

As the design evolves or requires changing, you can easily add or remove modules. Of course, in high-volume production, there is no replacement for a custom-built user panel that doesn't incur the cost of circuitry duplication of multiple modules.

IOSTRING kits are available at www.digitalcreationlabs.com/iostring.htm. You can build your own circuits with preprogrammed microcontrollers featuring the IOSTRING protocol. The flexibility of the kit format enables you to customize each board to contain only the required configurations. If you're running out of RS-232 serial ports, it's also possible to use any commodity RS-232-to-USB converter dongle to provide a USB interface.

You can use the IOSTRING system for a wide variety of applications. Industrial control panels, robot RC consoles, DMX dimmer consoles, virtual sound mixing consoles, and PC-based virtual oscilloscope control panels are a few examples. I plan to add a serial-to-Ethernet bridge such as a Lantronix XPort. This will Internet-enable the IOSTRING system.

Imagine flipping a switch in North America to turn on a light in China. Have fun, be creative, and unleash your inner dragon!

Listing 1—*Now you can use the IOSLIB host-side libraries. In this example, a single node of type IOS-IP-01 at address 1 is attached to the master computer. A structure variable is created to represent the single node in the network. This variable holds the node address, state information, and any registered function callback pointers. A series of initialization and registration calls are then made before the system is ready. The only other requirements are that a periodic call to the* `ioslib_protocol_process_msg()` *must be made for normal operation.*

```
ip01_struct ip01_node; //Structure variable for each node in string
#define IP01_PANEL_NODE_ADDRESS 1
                       //User-defined callback function called when
                       //button 1 is pressed
void ip01_node_cb_button1_pressed(void)
{
      printf("IP01 Button 1 pressed!\r\n");
}                      //User-defined callback function called when
                       //button 1 is released
void ip01_node_cb_button1_released(void)
{
      printf("IP01 Button 1 released!\r\n");
}
                       //Main loop
void main(void)
{
      ioslib_init();   //Initialize ioslib libraries
                       //Perform misc system initialization here...

      ioslib_ip01_node_init(&ip01_node); //Initialize structure
                       //Register callbacks before they can be used
ioslib_ip01_register_callback(&ip01_node,IP01_CB_BUTTON1_PRESSED,

&ip01_node_cb_button1_pressed);

ioslib_ip01_register_callback(&ip01_node,IP01_CB_BUTTON1_RELEASED,

&ip01_node_cb_button1_released);
      //Now, register node to activate it in the library

      ioslib_ip01_register_node(IP01_PANEL_NODE_ADDRESS, &ip01_node);

      for (;;)         //Main loop
      {
      ioslib_protocol_process_msg();  //Check and process any
                                      //messages in queue
                                      //Do other stuff here...
                       }
}
```

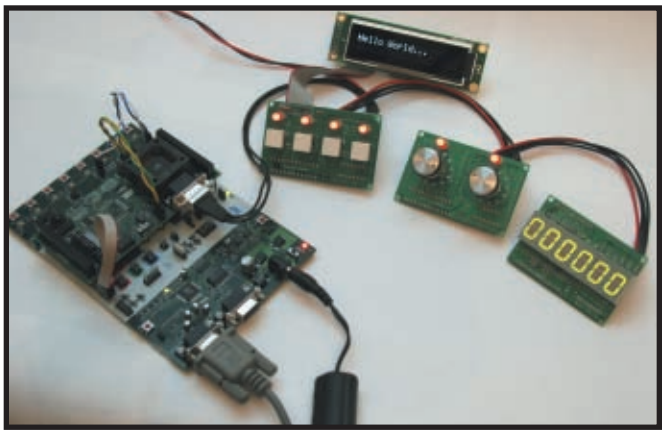


Photo 2—*For the experimental system, I used an off-the-shelf Atmel AVR STK500 development board (featuring an ATmega128L MCU) as the system host computer. The host controls three daisy-chained nodes consisting of an IOS-IP-01 board with an attached Optrex character LCD, an IOS-IP-02 board, and an IOS-OP-01 LED display board. In the future, all of these boards will be panel-mounted side by side in custom configurations.*

Eric Gagnon (M.A.Sc., P.Eng.) has been hooked on electronics since the age of 12. He earned both his Bachelor's and Master's degrees in electrical engineering from the University of Ottawa. With more than 14 years of embedded design experience, Eric has worked on projects related to the International Space Station, industrial robotics, and 3-D machine vision. He has also designed several commercial embedded video surveillance products. Eric currently runs Digital Creation Labs (www.digitalcreationlabs.com). You may contact him at egagnon@digitalcreationlabs.com.

PROJECT FILES

To download the code and additional files, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2006/196.

RESOURCE

IOSTRING Datasheets and message protocol definitions, www.digitalcreationlabs.com/iostring.htm.

SOURCES

STK500 Development board
Atmel Corp.
www.atmel.com

IOSTRING Kits and preprogrammed ICs
Digital Creation Labs, Inc.
www.digitalcreationlabs.com

Custom front panels
Front Panel Express LLC
www.frontpanelexpress.com

C Compiler for AVR
ImageCraft Creations, Inc.
www.imagecraft.com

Character LCD module
Optrex America, Inc.
www.optrex.com

Low-power PC/104 embedded platforms with ARM9
Technologic Systems
www.embeddedarm.com

Mini-ITX Mainboard
VIA Technologies, Inc.
www.viaembedded.com

34 CHANNEL LOGIC ANALYZER
500MHz Timing / 200MHz State at an *Unconventional* Price!
• Sequential Triggering on Edge, Pattern, Event Count, Group Magnitude/Range, Glitch, Duration and more
• +6V to -6V Adjustable Threshold supports a wide range of logic families including 1.5V LVCMOS
• Qualified State Mode and Selectable Setup/Hold Window offer very flexible high-speed sampling
• Real-Time Sample Compression enables capture of events separated by *seconds* with 5ns resolution
INTRONIX
LOGICPORT
34 CHANNEL LOGIC ANALYZER
pcTestInstruments.com
$389
Order Online Today!
• Compatible with USB 1.1 and 2.0
• Built-in 300MHz Frequency Counter
• Includes I²C, SPI and RS232 Interpreters
Download our free software and captured examples
INTRONIX www.pcTestInstruments.com Tel 602 493-0674 Fax 602 493-2258

Open-Source AVR Development

Ready to join the open source community? Colin introduces you to the AVR-GCC toolchain for AVR microcontrollers. Now you can contribute to a project from a Windows-based environment.

You've probably used an open-source tool like the GNU compiler collection (GCC). It's an open-source compiler that has been ported to countless different platforms and has even more targets that it can create code for. For the Atmel AVR line of microcontrollers, a perfect example is the AVR-GCC toolchain, which I'll describe in this article.

WinAVR provides a Windows-hosted distribution of the open-source AVR tools. It allows you to develop for the AVR from a purely Windows environment. Staying completely within the native Windows environment, you can make numerous small patches that will take only an hour to write and test. You can use your existing knowledge of AVR programming to get the job done.

Like everything in life, there is no end to what you can learn about open-source projects. Let's start with the basics.

AVR TOOLCHAIN

The open-source AVR toolchain is usually referred to as either WinAVR or AVR-GCC. Both are incorrect. Technically, one is a distribution of tools and one is a just a single project in the entire toolchain. Refer to Table 1 for a list of the tools and their functions. The toolchain typically starts at the compiler, which is AVR-GCC in Figure 1. This will turn your C code into files that can be executed by the AVR. However, the compiler knows only standard C. It doesn't know anything about the names of registers, the most efficient way to deal with floating-point numbers on the AVR, and similar problems.

If you have only the C compiler, you'll need to refer to each register by its address (not its name), write the start-up code, and much more! AVR-LIBC, which is the C library for the AVR, does this. It includes all the header files that contain the addresses of port and register names, the floating-point library, AVR-specific macros, and AVR start-up code. It also provides a lot of documentation on both the library items and a number of general items on the entire toolchain, including an FAQ.

The C compiler will generate assembler files for each input file (and have the assembler turn them into object code), but you still need just a single output file! The linker (avr-ld) will take all of these files and cross-reference function names between files to create one object file. The linker will also take modules from the C library and link them into the single object. Normally, this result is in ELF format, so an object copy (with AVR-OBJCOPY) is performed to generate a HEX format file. If any assembly sources are used, they're usually passed through the C preprocessor first and then passed on to the assembler (AVR-AS).

Note that these three programs are all contained in the GNU Binutils project, along with a number of other programs. The compiler, assembler, linker, and library form the core of the toolchain. Although the programs are interconnected, they're separate projects that are often developed together. Now you can use AVRDUDE to download to the target or AVR-GDB in combination with SIMULAVRXX or AVARICE to simulate or emulate the target device.

PROJECT WEB PAGES

Projects typically have a homepage and a project page. The homepage is a typical web site made entirely by people working on the project. It includes background information about the project, links to documents, information about the authors, and more. The project page is an automatically generated web site that's created by the project's host. In most projects, the host is either http://savannah.nongnu.org or http://www.sourceforge.net. You can normally get to the project page via a link on the homepage or by going to one of the hosts and searching for the project name.

The project page—which updates automatically and is often more up-to-date than the homepage—enables you interact with the project. Several tools are typically found on the page: links to downloads of file releases, the mailing list, repository information (Concur-

Tool	Description
AVR-GCC	C cross compiler for producing AVR code
AVR-AS	Assembler for producing AVR code
AVR-LD	Linker to link AVR object modules together
AVR-SIZE	Displays information such as the amount of flash memory or RAM taken by AVR object files
AVR-NM	Extracts symbol table from an object file, similar to a map file
AVR-OBJDUMP	Displays, extracts, or copies information from many AVR object file formats
AVR-LIBC	C runtime libraries, header files, and documentation for AVR target
AVRDUDE	Connects to programming hardware to download code into AVR
AVR-GDB	Debugger for the AVR target. It needs a backend such as simulator or emulator.
AVARICE	Connects to the JTAG-ICE (AVR emulator) and allows it to work with avr-gdb
SIMULAVR	Provides a simulator back end for avr-gdb

Table 1—*The AVR toolchain label includes many different tools. This table summarizes the most important tools and the ones you are most likely to run across. Note that some of the tools are part of larger projects. Others are entire projects. Binutils encompasses AVR-AS, AVR-LD, AVR-SIZE, AVR-NM, and AVR-OBJDUMP.*

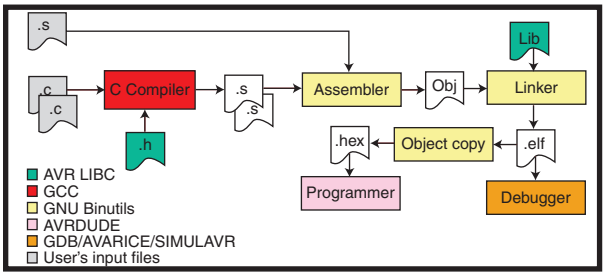


Figure 1—*Study how a user file moves through the open-source AVR toolchain. It isn't specific to the AVR processor, but the tools are different for each target. The makefile, which includes the settings for the various tools, isn't shown.*

rent Versioning System (CVS) or Subversion (SVN)), a bug tracker, a patch tracker, a feature tracker, and news.

The files section includes links for downloading official releases of the project. An official release typically means that a collection has been tested and given a version number. The mailing list contains information about subscribing to mailing lists for the project and an online archive (or previous posts) that you can search through. Project development occurs in the repository. You can take snapshots of all the files in their current state.

The three trackers provide an easy way for users and developers to interact. Anyone can add an item to these trackers. You can specify an item's importance, add files, and discuss the item. Note that the trackers are not general-purpose discussion boards. If you're unsure about adding an item, ask the mailing list! The trackers make sure an item isn't forgotten. If you want to contribute to a project, you can check the trackers to see what needs to be done without having to go through all of the source code.

The bugs tracker is for bugs found in the project. When adding a new item, always check with the mailing list unless you're completely sure it's a bug. (You don't want to make any mistakes.) A good bug report is always useful to developers. Such reports should be attached to the correct tool (e.g., AVR-LIBC, GCC, and Binutils). You must list the minimum code required to demonstrate the bug. You should also search the bug tracker for your specific bug. Be sure to search for closed bugs too. You never know when someone will fix a bug.

You can upload a new patch for the project in the patches tracker. This isn't the place to upload a patch that fixes a bug. Those go in the bug tracker with the specific item. For example, if you decide to add support for a new processor, you can add the patch to the patch tracker.

Finally, general statements about the project's future are made in the feature request tracker. This is also called the task tracker (depending on the host). In this tracker, you can discuss how to get a certain item done and its status.

The project page is more than just a simple web site or collection of features. It is generally the heart of a collaborative project. The page holds all of the tools for the project in one spot and encourages interaction between them. When discussing a problem on the mailing list, you can refer to the bug report number back on the web page. When someone adds a new patch to the bug, an announcement is automatically sent to the mailing list. When the project releases a new version, a news release is posted on the project site. (At times, a release is also sent out to a list of subscribers.) These file releases can be mirrored across the world so anyone can access a fast local mirror of the file. You can sign up with a specific account and become a member of a project. Members have varying degrees of access to information. Some members can only update bug reports. Others have the administrative responsibility of managing other members.

TALK TO EVERYONE

When working on a collaborative project, you need a way to interact with other members. Almost all open-source projects have a mailing list for this task. A mailing list connects everyone who is signed up. There is a central address to which you can send a message. After you send the message, it's distributed to everyone on the list.

A mailing list is better than an online forum because your messages are delivered to your e-mail client. Most clients can filter mailing lists, which means all of the messages from that list are delivered to a folder. As a

TRI-M ENGINEERING

PC/104 Multitech Socket Module Carrier Board



FLEXCOM104

Two serial ports (RS-232/RS-422)
Connects either serial port to ethier Universal Socket in Standalone mode
Allows bridging of the modules
PC/104 interface to high speed quad UART

100Mhz PC/104 Module



MZ104

Featuring the new edition ZFx86 FailSafe® Embedded PC-on-a-Chip
Dual watchdog timers, Phoenix BIOS and FAILSAFE Boot ROM
Extended temperature -40°C to 85°C

168 Watt High Efficiency Power Supply



HPS3512

Total Power: 168 Watt with ATX Interface
+3.3V, +5V, 12V output
6V to 40V DC input range
Extended temperature: -40°C to +85°C
RS232 serial port / Optocoupled inputs

www.tri-m.com info@tri-m.com
1.800.665.5600
HEAD OFFICE: VANCOUVER
tel: 604.945.9565 fax: 604.945.9566

result, you don't have to check different forums to keep up-to-date with a number of projects. It's all in one program.

If you are new to the system, it's always a good idea to subscribe to the mailing list and watch it for a few days before posting. This is a great way to learn how it works. As a result, you can view and search the archive of the mailing list online. Always search the archive before posting a new question!

Listing 1—*The lines with a minus sign (–) in front of them are present in file1.c but not file2.c. The lines with a plus sign (+) are present in file2.c but not file1.c. Lines with nothing in front of them are common to both files and are used to show the context of the changed lines.*

```
--- file1.c      2006-04-02 17:13:27.000000000 -0300
+++ file2.c      2006-04-02 17:13:36.000000000 -0300
@@ -5,7 +5,7 @@
    int counter;
    char received;

-     while (received = 'c')
+     while (received == 'c')
    {
    received = get_char();
    }
```

DEVELOPMENT ENVIRONMENT

All of your work is performed in the development environment on your operating system, although it is easier to do this from a UNIX platform. You can use Linux, Mac OS X, or FreeBSD. Most Linux distributions are set up with all of the developer tools you need, although they don't come with the toolchain for the AVR because it's so specific. Building the toolchain from source code is a good idea if you plan on contributing to it because it gives you an idea about what is involved in each project. In addition, when you start modifying the source code, you'll need to build the tool to see the changes. It's good to start with code that you know works.

If you're using a Windows-based system, your development options are somewhat more limited. You can install WinAVR, which is a collection of .exe (binary) releases of the AVR toolchain. You won't be able to compile your own AVR toolchain with this, which means you can't fix any bugs that occur in the source code or try your own modifications. However, you can still contribute to the development of AVR tools with just WinAVR installed. You can fix errors in the header files provided with avr-libc. You can also add support for new parts to AVRDUDE and a few other little tasks.

If you choose to do complete development in Windows, you must install Cygwin. This provides a Unix-like environment for Windows. When installing Cygwin, be sure to specify that you want the developer tools. This will install GCC, Make, Autoconf, and a few other things you'll need. After you have Cygwin, you can follow the directions in the AVR-LIBC online manual for Unix-like environments for building the tools.

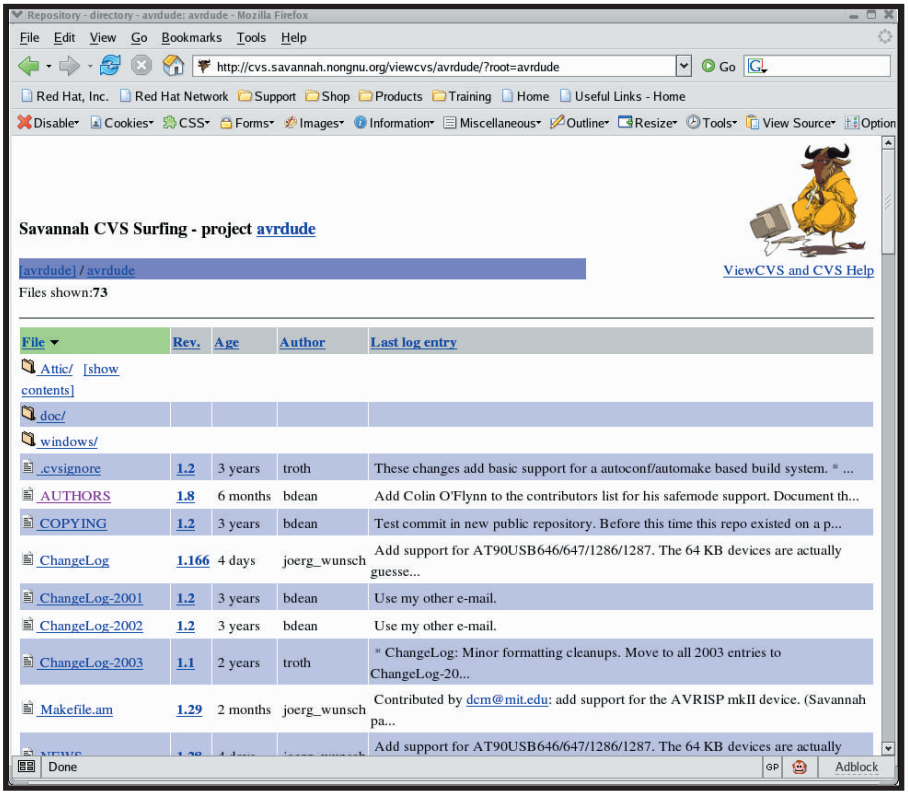


Photo 1—*Take a look at a CVS viewer on Savannah. You can see the file name, the version, the last modification, when the modification occurred, and who performed the modification. Clicking on a file enables you see a list of all the available versions and the differences between any two versions. All of this is done with a graphical interface via your web browser. It's color coded just like your editor.*

PATCH

Patching source code is a very important concept in open-source software. Luckily, it's a simple one. When you have a file that needs changing, you must be able to quickly determine which changes are needed. If the change breaks something, having the changes listed will make it easy to pinpoint the problem. It's easy to see when entire sections have been inserted, but it's often the little changes that are easy to miss that create problems.

Use the "diff" and "patch" software. The former takes the difference between two files and outputs a readable file. Let's say you have two files: file1.c and file2.c. You can compare the differences:

```
diff -u file1.c file2.c
```

Listing 1 shows an example output with a description of the lines. In the real world, file1.c would be a copy of the original file, file2.c would be the file you changed, and the output would be your changes. You can pipe this

output to a file instead of the screen:

```
diff -u file1.c file2.c > output.diff
```

This results in an output.diff file with the changes in it. This output file can be used as the input to the patch program. Patch reverses the operation of diff by taking a patch file to create an updated file.

To convert file1.c into file2.c, type:

```
patch -i output.diff
```

This looks in the current directory for the first file specified in output.diff (file1.c in this case). It then applies the changes to it. The result is a file1.c that is the same as file2.c.

If you have directory names to worry about (such as a large patch file), you must use the -p option. The -p0 option uses the directory names as they are given in the patch file. Every "1" after that removes one level of slashes from the directory. For example, if the patch file has file1.c in tmp/file1.c and you have it in the main directory, you can type:

```
patch -p1 -i output.diff
```

This will remove the "tmp" from the path name before applying the patch. You might need this option if patch complains that it cannot find the requested file. Take a look at the patch file to see if the file was created a directory or two down from where you are.

The patch process even works when applying the patch to a file that has changed slightly from the original. It will attempt to find the location of the change in the new file if there is a simple offset.

VERSION MANAGEMENT

Almost all open-source projects use some sort of version control software. This software allows you to manage multiple people working on a project. It also enables you to stop someone (or an error) from destroying other people's work. It does this by keeping a history of all changes and making older versions of a file available.

The tools currently in use are CVS and SVN. You can view the files and different versions online to quickly see the changes (see Photo 1).

A repository tool manages the source code. You can do a number of operations to the repository. As an unprivileged user, you can "check out" files by copying a file or files to your computer. You can also update your local or working copy of the files so they are the same as those in the repository module. Local changes are preserved by the update operation. You can also perform a "diff" of your local files to CVS, which can be used for creating a patch file.

These tools work with the previously mentioned diff and patch tools. You can request diffs between any two versions to see all of the changes. You can also create a patch against CVS by creating a diff from your local files compared to the files in CVS.

After you get the hang of CVS and submit a few patches to the project, you can often get member CVS access. With access, you can now commit your changes directly to CVS. On the project page's CVS page, you'll find a link to CVS documentation and examples to get you started.

WHO SPILT BEER ON ME?

If you are new to the process of contributing to open-source projects, expect to commit a few party fouls. But you can definitely avoid some of them.

Line ends can be a problem. Windows tends to use both a line feed and character return (CRLF), whereas UNIX systems use only a line feed (LR). If you use a text editor other than Notepad, there should be an option for saving line ends. Always try to use the UNIX style.

Another common source of problems is the mailing list. E-mail is often a poor medium for communication. You can misunderstand a message and feedback isn't immediate. Don't get upset if you don't agree to something and always think twice before replying. Try to consider the other person's position first. Also, keep in mind that many people don't like to write long responses, so don't be surprised if you receive a terse message.

PART OF A TEAM

One problem with open-source projects is that there is no easy way to thank everyone involved. Some projects involve so many people that you end up with a list that's longer than your original article!

I would like to thank Brian Dean, Eric Weddington, and Joerg Wunsch for their assistance with this article. They helped me get started in the open-source world. They explained where I went wrong with CVS commits, patches, and mailing lists.

Brian originally wrote AVRDUDE to address the lack of an AVR flash memory download program for his favorite development platform (FreeBSD Unix). Several years ago, he released the source code to the public to facilitate development and porting to other platforms. Since then, many people in the open-source AVR community have made contributions.

Open-source projects are also a great way to get noticed in your field. Good luck. Perhaps we'll find ourselves contributing to the same project.

Colin O'Flynn is an electrical engineering student at Dalhousie University in Halifax, Nova Scotia. Despite being a full-time student, he enjoys electronics enough to spend extra time working on electronics projects and contributing to open-source AVR projects.

RESOURCES

K. Fogel and M. Bar, *Open Source Development with CVS*, Paraglyph Press, 2003, http://cvsbook.red-bean.com/.

Free Software Foundation, "Comparing and Merging Files," 2002, www.gnu.org/software/diffutils/manual/diff.html.

———, GNU Manuals, www.gnu.org/manual/manual.html.

R. Neswold, "Installing the GNU Tool Chain," www.nongnu.org/avr-libc/user-manual/install_tools.html.

E. S. Raymond, "Asking Questions The Smart Way," 2006, www.catb.org/~esr/faqs/smart-questions.html.

SOURCES

Cygwin
www.cygwin.com

WinAVR
http://winavr.sourceforge.net

Electronic Gear Control

Add Electronic Gears to a Metal Lathe

John upgraded his metal lathe with a PIC18F2620-based gear control system. In this article, he explains how to design and write code for your own electronic gearbox.

Eleven years ago, I stumbled across Dave Gingery's *Build a Metalworking Shop from Scrap* series. I'd always wanted a metalworking machine shop to augment my woodworking tools, but I hadn't been able to justify the costs. Reading Gingery's books seemed like the easy and inexpensive way to get into metalworking. I found his address on the inside cover, and a short while later I had the seven-book series, which covers the processes of building a charcoal foundry, a metal lathe, a shaper, a horizontal mill, and a drill press. The sixth book describes a series of attachments for the lathe and the creation of a dividing head that enables you to make gears and add the screw-cutting capabilities to the lathe. In the Threads sidebar (p. 38), I explain why gears are needed and how a lathe with change gears can cut threads.

During this period of discovery, I also learned that Gingery had designed a gas-fired foundry furnace and that his son, Vince, had designed a power hacksaw and slip roll. It seemed like the correct sequence would be to build the hacksaw and slip roll and then use them to make the gas-fired foundry furnace. When the furnace was done, I figured I'd start on the machine shop tools. I went one step further and dug through my old copies of *Circuit Cellar* and found David McFalls's 1993 article on a computer-controlled kiln/furnace ("Fire Control for Foundry Furnaces," *Circuit Cellar* 39). What I decided to build, of course, had to have an embedded computer too!

After years of on-again, off-again construction, pattern making, filing, and drilling, I finally built a working metal lathe built from scrap aluminum and cold rolled steel. Although the Gingery lathe is functional for simple turning, I never managed to find the time to finish the shaper and horizontal mill. That meant the gears for screw cutting were difficult if not impossible to make. As a result, I used a threading die whenever I needed threads for parts made on the Gingery lathe, but the threads weren't always straight. It was time to either make gears or add CNC using a PC running ArtSoft MACH 2 software. But I really couldn't see a PC, screen, and keyboard connected to the small Gingery lathe. My woodworking and metalworking shop share the sawdust, and I was running into serious space issues. What I really wanted was a set of gears and perhaps a digital read-out (DRO) that could tell me the carriage's location at any time.

It was at that time that an old 10" Southbend lathe crossed my path and followed me home while I wasn't looking. As a result, my work on the Gingery lathe stopped until a series of postings in the Yahoo! CADCAMDRO group restimulated the idea of gears (of the electronic variety) for the lathe. It seemed like a great idea, but not really CNC, so I joined the newly formed E-Leadscrew group.

COLLECTIVE DESIGN

After a lot of discussion, the group came to the consensus that a computer-controlled lead screw to replace or augment gears for adding threading capability would be useful if it were both inexpensive and small. And, as usual for a group of creative individuals, there was both tension and drama as we argued about how many features such a product would have. Some wanted a minimal unit with two knobs and a button (and an LCD), while others wanted a full set of features, a large LED display with a huge number of buttons, and two-axis control for cutting tapered threads and automating the threading process.

We also had numerous discussions about controlling a stepper motor or servomotor and its accuracy. Our initial guides were articles by Fritz Linck in which he described an electronic lead screw (ELS) using CMOS digital logic, multiline encoders on the spindle, and the lead screw, all of which were configured with a PLL circuit with dividers and multipliers to link the DC-servo-driven lead screw to the spindle. This PLL approach used a multiline encoder on the spindle to increment a counter, while a second encoder on the lead screw (driven by a DC servomotor) decremented the same counter. If the lead screw's speed exactly tracked the spindle, the counter would always be zero. A positive or negative difference drove an error signal that controlled the speed of the DC servo. Synchronizing the two so that the thread would start at exactly the same point each time would still require an indicator gear, and I couldn't see how adding a digital readout of the carriage position would be easy. I decided it would be less expensive and simpler to use a small microcontroller and stepper motors.

PROTOYPE HARDWARE

As it happened, I had a couple of Microchip Technology PIC boards

with a small prototype area left over from another project (see Figure 1). I knew that these would be perfect for research and development.

The board I used included a PIC18F2620 microcontroller. I wrote the code in C language with Microchip's MCC18 compiler. The prototype area had room for a couple of connectors and an LM339 used as a comparator with hysteresis to get rid of noise pulses at the slotted disk (see Photo 1, p. 40).

My Gingery lathe has a half nut and a handle on the leadscrew on the right side of the carriage. Power feed is accomplished by stretching a round rubber vacuum belt around pulleys on the left-hand side and engaging the half nut. Power feed is only available in the direction of the headstock. I removed the lead screw handle on the right and replaced it with a toothed belt pulley and mounted the stepper motor in the rear (see Photo 2, p. 41). No need for the O-ring belt on the left-hand side. I also fabricated a disk with a mounting ring and cut a small slot in the disk. This was slipped on the spindle beside the four-step pulley. The slotted optical sensor was mounted so the disk and slot created a stable waveform (see Photo 3, p. 41).

When this gap in the disk went through the optosensor, it caused an interrupt and latched a 25-kHz pulse clock counter. I now had a method for determining the spindle's RPM and also a repeatable spot on the spindle. Each interrupt told me where I was and how much time had passed since the last interrupt. With this information, I could start the lead screw at exactly the same point and at the correct speed.

```
RPM = (PULSE_CLOCK_RATE * 60L)
  /PulseClocks;
```

I also wired up two small buttons to use as jog buttons for tweaking the carriage's position in preparation for threading. All of the other parameters and commands were done from the PC via HyperTerminal. Here the goal was to get the PC out of the workshop, and yet I needed it for configuration. "No matter," I told myself. This was a prototype for research and development.

A project or two prior to this one, I had also modified and upgraded the floating-point routines from Daniel Ramirez's 2004 article entitled "Math Coprocessor for Robotics Applications" (*Circuit Cellar* 172). So now I was armed with both 32-bit integers and floating-point for a very inexpensive but powerful processor. The approach I'd used a few years earlier on a PIC16F877 still seemed ideal, so I ported the stepper motor code to the MCC18 compiler, and within a short time I was able to make the code turn the motor. Problems with accurate speed control immediately surfaced. These were the same problems I had experienced a few years earlier.

Low speeds are really easy. Simply program one of the compare registers to output a clock pulse based on dividing down the system clock. The trouble was that the faster I wanted the

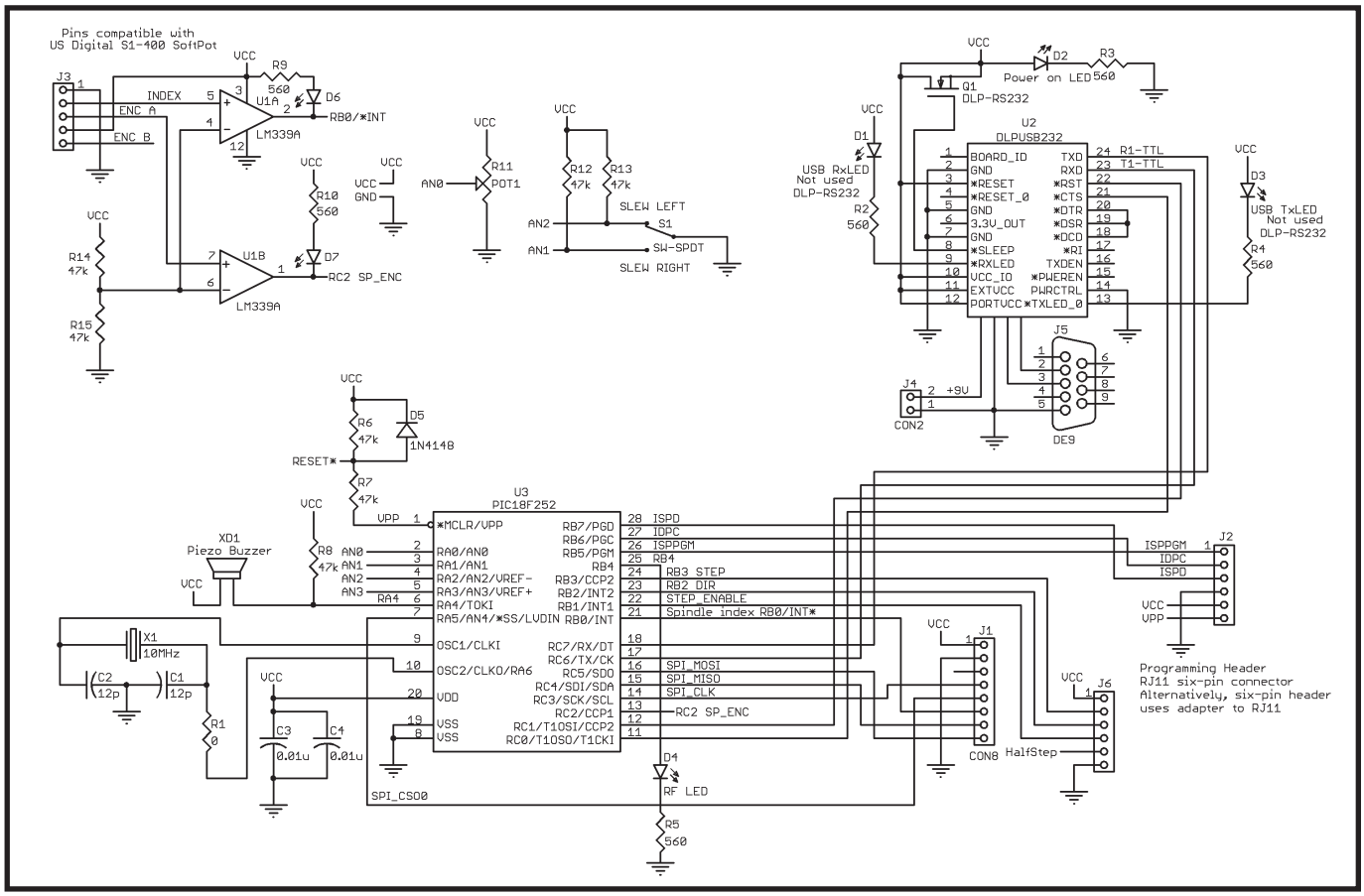


Figure 1—*Take a look at my hand-wired prototype. This PIC18F2620-based version used a serial port for command and control. The step and direction were connected to a microEngineering Labs stepper driver.*

motor to turn, the larger the steps between frequencies. For example, 1 MHz divided by 50 is 20 kHz. If I wanted 20.5 kHz, I would have had to divide by 49 and settle for 20.408 kHz. It would have been close, but it would have changed the exact ratio that I needed between the spindle and the lead screw. It also meant I would have had to use a second counter module to keep track of how many step pulses had been issued. Accelerating and decelerating would have added even more complexity, so just using the internal counter/timers in that way wasn't going to work.

My solution was to set up a 25-kHz clock and interrupt on each 40-µs pulse clock. After *n* of those, I could issue a step pulse and increment the number of steps issued. The problem was that my motor speed would jump from 25 (no pulse clocks between steps) to 12.5 kHz with one pulse clock between steps. It didn't matter for slow speeds, but it was a serious issue with microstepper drivers that could run from 10 to 25 kHz.

ONE STEP BACK

When it came to electronically connecting the lead screw to the spindle, I was still influenced by the PLL concept of controlling the lead screw by just monitoring the speed of the spindle and creating an exact lead screw speed based on the desired ratio. Obviously, I couldn't just arbitrarily pick an exact lead screw stepper motor

Threads

Most physics textbooks describe a thread as an inclined plane that travels around a cylinder. Threads can be produced by rolling, grinding, or cutting. Rolled threads are formed by squishing the metal into the shape of a thread with dies and a lot of force. Ground threads use a precisely shaped grinding wheel to form very accurate threads. Cut threads are the ones you are probably the most used to. They are often made with a die.

The die is like a nut, but it has hardened threads that act as multiple teeth with gaps (to leave room for the chips that are carved away). Starting the die so that the thread is formed evenly around the cylinder can be extremely difficult, but the threads cut by a die are very clean without much tearing.

On a lathe, a single point tool, ground to the exact shape of the thread, is used to cut the groove in the object being turned. In order to avoid tearing up the metal, the thread is cut using multiple passes. Each pass removes a tiny bit more metal until the thread is deep enough to meet specifications.

Each thread type is defined by its form and pitch. Most of the common threads in use today are American Standard Unified coarse pitch, fine pitch, and metric. These threads are triangular with a 30° angle. Pitch is defined as the number of inches or millimeters between threads or as the number of threads per inch. There are a number of standard thread pitches like 20 threads per inch, which is a 0.050″ pitch. That means each peak (or valley) of the thread is 0.050″ apart. A metric thread is usually defined by the pitch in millimeters.

The metal lathe uses a threaded shaft called a leadscrew to move the carriage and the cutting bit along the length of the bed. The leadscrew's pitch determines how far the carriage will move for every revolution of the leadscrew. If the leadscrew is 10 TPI, then it moves the carriage 0.100″ every revolution. Cutting a 10-TPI thread means that the gearing on the lathe has to be configured to turn the leadscrew one turn for every turn of the spindle. A 20-TPI thread is cut when the 10-TPI leadscrew turns a half turn for every turn of the spindle.

Metal lathes with threading capabilities often come with a chart and a set of gears that allow an operator to set up the common threads and a number of very fine pitches for turning the work (as little as 0.0015″ movement per revolution of the spindle). If you're lucky enough to own a lathe with a gearbox, you don't have to get dirty fingers handling greasy gears. You don't have to do any fancy calculations either. Just set the levers to positions based on a pictorial chart and cut away. Metric threads on an imperial lathe require a transposing gear to convert the 25.4 mm/inch ratio. Usually, it's a 100- and 127-tooth gear in the reduction drive train that performs this division.

After setting up the gearing, the tool bit is moved to the correct position and a lever that engages the half nut on the lead screw is actuated to start carriage movement. In order to know when to engage the nut, a special lead screw gear driven indicator is watched. When a number on this turning indicator dial lines up with a fixed mark, the half nut must be engaged. Without this indicator, the tool bit might be offset and the next cutting pass will not line up with the previous pass.

At the end of a pass, a number of things have to happen very quickly. First, the half nut must be disengaged before the tool bit starts cutting past the end of the thread. In fact, it could run into a shoulder and something might break. The tool bit must be withdrawn, the carriage must be moved back to the starting position, and the tool bit must be moved back into the work. During each pass, the tool is inserted a bit deeper in order to cut away a more metal. Finally, for each pass, the half nut must be engaged at precisely the same point in order to line up the cutting tool correctly.

For metric threads on an imperial lathe, the half nut must never be disengaged. The leadscrew must be stopped and reversed so the carriage is returned to the starting point. Things will then proceed the same way. Adjust the cutting depth, start the spindle, and cut the next pass.



CONTROL MADE EASY

Need to monitor or control the small sensors and actuators in your control network? Or make your machine more reliable, easy to assemble and remotely accessed? Use the Pyxos™ platform, the embedded control networking solution that installs with push-button simplicity. Pyxos chips are small enough for nearly any sensor or actuator, and includes a digital I/O and SPI port to use with existing microcontrollers. It extends the power of LonWorks® control networks to the smallest of devices, plays well with others—host processors and PLCs—and has the development tools and reference designs to reduce your development time.



- ☑ Deterministic TDMA protocol
- ☑ Bit rate: 312.5kbps
- ☑ Scan time for 32 devices: ≤25ms
- ☑ Wire type: 16AWG (1.3mm) to 24AWG cat5 (0.5mm)
- ☑ 24VAC or DC power and data on two wires
- ☑ Application power delivered to each device: 100mA@3.3VDC
- ☑ Built-in unique MAC ID
- ☑ Plug-and-play or preconfigured self-installation
- ☑ 20MLF (5mm x 5mm) RoHS-compliant IC package

Discover the technology that millions of devices worldwide already use.
Call 1 - 408 - 938 - 5200 or visit us online at www.echelon.com/easycontrol.

ECHELON®

© 2006 Echelon. Echelon, LonWorks, and the Echelon logo are registered trademarks of Echelon Corporation registered in the United States and other countries. Pyxos is a trademark of Echelon Corporation.

speed. It was time for some reverse engineering.

The MACH CNC software can run the stepper motors at speeds up to 25,000 pps with nice accelerations, and it can track the spindle for threading. Even more interesting is that it does this with one pulse per spindle revolution using one optical sensor and a disk with a single slot. A single pulse encoder was easier to implement and less expensive than a multiline encoder.

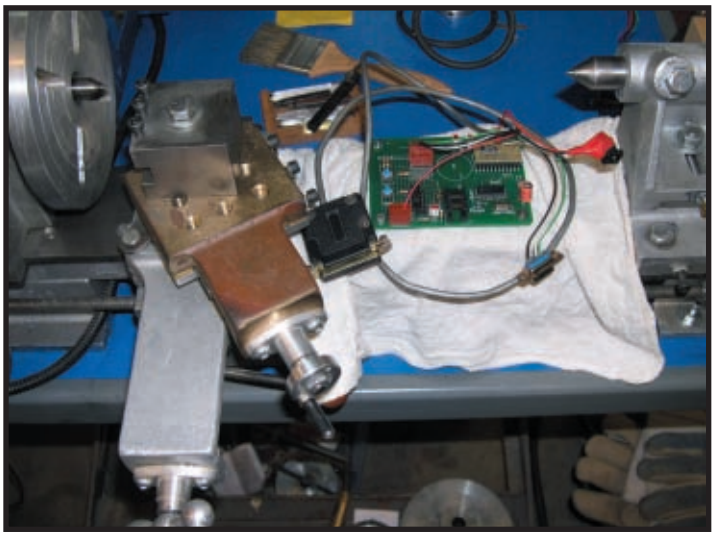


Photo 1—*Originally designed for a DLP-USB232, the PIC18F2620 board has a small RS-232 converter and 5-V regulator. The prototype area has the comparator and connectors for the stepper motor driver and the spindle encoder. The board is only 2″ × 3″.*

Something had to be happening in the 40-µs pulse clock time period that didn't need fancy comparators, which was important because the PC doesn't have any. But this was an 8-bit processor, so I didn't want to do too much in the way of 32-bit integer operations either, which are terribly expensive on an 8-bit processor.

The solution came from several different places. I e-mailed a friend with a description of the ELS project. He suggested using the Bresenham algorithm for drawing lines in two axes. After referring to the graphics books, I realized the technique I had worked out was a modification of that algorithm. Great. Two minds think alike. Well, three minds actually. Another clue came from my copy of MACH2 software. More on that later.

To draw a two-dimensional 45° line, the Bresenham algorithm requires a step in the x direction to have one in the y direction. Drawing a line on a different angle may require three x steps for every y step. The Bresenham uses floating point (or something like it) to do these calculations. Let's say I want to display a line that has the y move one pixel up for every four pixels over. To put it another way, for every x pixel, y moves 0.25 pixels. For each x, the y is accumulated in a register. When y is greater than or equal to 1.0, a physical y movement is made.

I downloaded a spreadsheet with many of the basic calculations required for determining the lead screw's speed based on pulse clocks per revolution. I used the spreadsheet as a template for developing some code. But instead of a compile, flash, and test cycle on the PIC, it was easier to whip up a Delphi 5.0 simulation that allowed me to enter in the various lathe parameters and then look at the variables at each 40-µs interrupt.

I could now write the C code for the PIC microcontroller based on what I'd learned from the simulation because it turned out that driving the stepper motor at a specific speed wasn't going to be that difficult. Knowing the number of pulse clocks per revolution for a specific RPM, the number of motor steps per revolution, the lead screw pitch, and the target thread gave me a ratio that I rounded to four decimal places. I didn't want to use floating point in the interrupt routine to represent the number of steps to turn, so I used two 16-bit integers: one for the integer part (Istep) and one for four fractional digits (Fstep). The fractional integer was a scaled number from zero to 9999. I represented this number as Istep.Fstep.



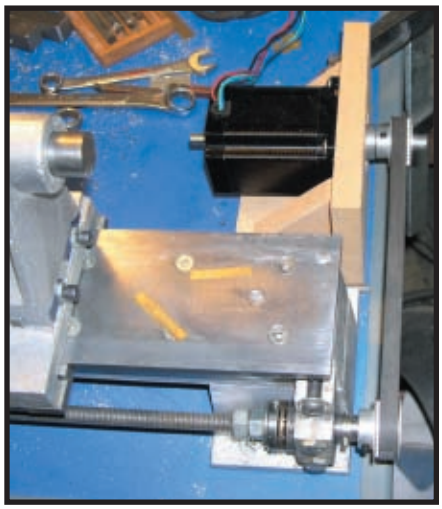


Photo 2—*The stepper motor is connected to the leadscrew with a toothed belt to avoid any slippage. The ratio is 1:1 because this motor has more than enough torque to cut most threads at the slowest spindle speed of 275 RPM. A 1:2 ratio would double the speed of the leadscrew and allow coarser pitch and double start threads.*

For each 40-µs interrupt, I decremented the integer part of the divisor. A step pulse was generated at zero. At each step, I also added the fractional part to a running total counter. As soon as it exceeded 10,000, I incremented the step pulse counter and subtracted 10,000 from the counter (see Listing 1, p. 42).

NUMERICAL EXAMPLE

You can cut 18-TPI (threads per inch) threads at 274 RPM with a lathe that has a 16-TPI lead screw. This means the lead screw has to turn 0.8889 revolutions per spindle revolution. The lead screw must turn 243.58 RPM.

If the spindle is driven at 274 RPM, there will be 5,474 pulse clocks per revolution. The stepper motor connected to this lead screw has 200 steps per revolution.

$$\text{Lead Screw Ratio} = \left(\frac{16}{18}\right) \times 200 = 177.7778$$

$$\text{PeriodDivisor} = \left(\frac{5{,}474}{177.7778}\right) = 30.7912$$

$$\text{Istep.Fstep} = 30.7912$$

The 25-kHz interrupt routine has to count down 30.7912 of the 40-µs pulse clock periods and step. That's what makes the stepper motor run at 243.58 RPM.

Now you can see how the Bresenham approach is used. I count down 30 interrupts, step the motor, and add 30.7912 to the IstepCounter.FstepCounter pair. The integer part of 31.5824 (the new result) is the next value that the interrupt routine has to count down to zero and then step. After that, the next value becomes 31.3736 (i.e., 30.7912 + 0.5824) and so on. This is done with 16-bit additions inside the interrupt routine after the initial floating-point calculations are complete. Listing 2 (p. 43) shows how I handled distance, acceleration, and deceleration.

BACKWARDS FORWARD FASTER

Stepper motors can almost instantly accelerate to a specific speed. If you try going faster, the rotor will lock and it won't move until the step rate is decreased. At higher speeds, stepper motors also have much less torque. But higher speeds are possible if the motor is slowly accelerated up to the target velocity.

I was already counting 25-kHz pulse clock periods, so it seemed simple enough to just increase the `IstepCounter` variable by some value that would slow down the motor by increasing the time between steps. At each step event, I decremented this extra acceleration value until it was zero and the motor was traveling at full speed.

Deceleration happens the same way, but the time between pulses increases. The acceleration value is also used as a step counter. This value

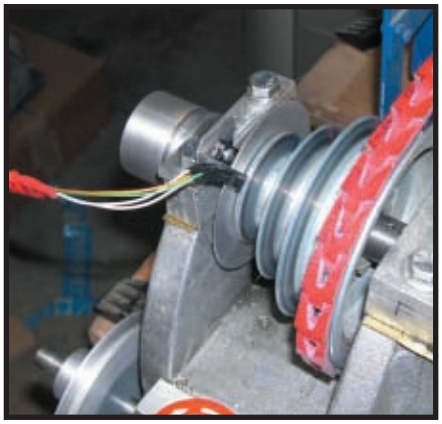


Photo 3—*A slotted optical sensor salvaged from an old printer was put into service as the spindle speed and position sensor. The slot in this case is a small hole drilled in the disk that isn't visible because the hole was out of sight when the flash captured the picture.*



is also used to track the number of steps left in the move before the period between step pulses lengthens again.

At that point, I put some cold rolled steel between centers and cut a clean 24-TPI thread. Under a 10× magnifier, the quality of the threads matched what MACH2 Lathe software created on the same lathe with the same stepper and spindle index circuit. But I noticed that I was never able to make my stepper driver code run the same motor as fast as the MACH software. Something remained different, but what was it?

Listing 1—*The original interrupt routine shows how spindle speed, movement distance, and stepper acceleration and speed are controlled.*

```
LeadScrewRatio = (LeadScrewTPI/ThreadTPI) * MOTOR_STEPS_PER_REV;

PeriodDivisor = PulseClocksPerRev/LeadScrewRatio;

Istep = PeriodDivisor; // Automatic truncate.

// Round up and create fractional part * 10000.
Fstep = ((PeriodDivisor + 0.00005) - Istep) * 10000;
```

CNC CONNECTION

Because I had a working project, I figured it wouldn't hurt to send a code fragment to Art Fenerty, the author of the MACH CNC software, and ask for some feedback. He was a great help. Art looked at my code and offered a fragment of his own JOG code from the MACH2 device driver. It turned out that we were doing virtually the same thing, except my code was based on counting periods and his used frequency. It was obvious just how much could be done with at least a 1-GHz Pentium III, 32-bit registers, and a barrel shifter.

I always enjoy a challenge, so I tried porting the MACH PC code to the PIC18F2620. My only worry was processing speed. In order to have processing time for the other parts of the application, I didn't really want the interrupt code to take longer than 20 µs.

Let's consider how the JOG software worked. Instead of using two 16-bit integers to hold an integral and fractional part, a 32-bit integer held the fractional part in the first 11 bits and the integral part in the rest. The fraction could range from zero to 2,047. When it increased beyond that, it automatically incremented the integer part and wrapped around through zero.

The integer part was limited to the maximum pulse clock value of 25,000 Hz. Each time the integer part added to a `holding` variable overflowed by 25,000, a step was sent to the motor. The `holding` – 25,000 was the new Bresenham accumulator value.

The `MaxVel` set point is the stepper motor speed to the nearest hertz. There's a downside for an 8-bit PIC. During acceleration and deceleration, the acceleration value is added to the velocity counter as a 32-bit addition operation. To determine whether or not to step, the fractional part is stripped off with an 11-bit shift and added to the holding accumulator, all of which takes multiple PIC clock cycles but only one machine cycle on a Pentium.

When I was done, I scoped the digital flag bit that I had set and cleared inside the interrupt routine. I found that the overall time in this algorithm was almost twice as long as the 16-bit version, but it was still just within the design specifications. The motor (controlled with a 10-microstep driver) accelerated to 25,000 Hz (750 RPM). At that speed, with my motor and drive combination, I could slew the carriage at 47 IPM, but I'd skip steps while cutting due to a lack of torque. Without a back gear to reduce the spindle speed, I was therefore limited to cutting threads greater than or equal to 10 TPI. I was a happy camper (see Photo 4)!

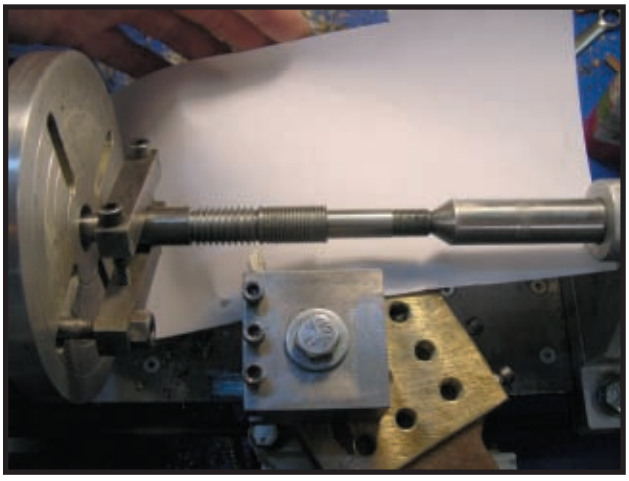


Photo 4—*The fine pitch metric 1-mm pitch thread on the far right fits perfectly in my Unimat tooling. An imperial 18-TPI thread is in the middle. A coarse 10-TPI thread is on the far left. All are cut using the ELS software on the Gingery homebuilt lathe.*

UNFRIENDLY USER INTERFACE

The code posted on the *Circuit Cellar* FTP site shows the wide range of commands I built into the system. The code enables me to not only debug, but also use the system as an ELS. I quickly discovered that the command-line-oriented user interface wasn't suited for running this type of machine.

For some commands and features, it really didn't matter because setting fixed parameters like leadscrew pitch and motor steps per revolution were entered once and then never touched again. Even the desired thread was usually set only once and then not touched until a new thread was needed. But actually cutting threads involved a number of keystrokes, and it was easy to make mistakes. For instance, I'd start the threading from home (0.000") for a given distance rather than from a start point to an end point. As a result, if I miscalculated one of the two positions, I'd end up with a damaged project or a broken tool. Don't ask how I know that.

MOVING FORWARD

The original purpose of this project was to develop an inexpensive, compact unit that wouldn't be larger than the lathe it was controlling. Spending extra time on a serial port user inter-

Listing 2—*The original interrupt routine shows how movement distance and stepper acceleration and speed are controlled. It also shows how the spindle sensor interrupt is translated into counts per revolution.*

```
void __INTH(void) {
if (PIE1bits.CCP1IE && PIR1bits.CCP1IF) {
      PIR1bits.CCP1IF = 0;
    bSTATUS = 1;          // Check how long this takes.
                          // Another pulse clock TICK.
    SpindleClockValue++;
    if (fMoveBSY && (StepCount != 0)) { // Not finished stepping
                                        // the distance yet?
        // Enough integral pulse clocks occurred which means we
        //  send out a step pulse.
      if (--iClocksPerStepCounter == 0) {
          bM1StepMotor = 1;  // STEP
    // Track how many pulse clocks we have to wait util we step again.
          iClocksPerStepCounter = iClocksPerStep;
          // Integer math for speed inside interrupt routine.
            fClocksPerStepCounter = fClocksPerStepCounter +
          fClocksPerStep;
          // overflow 0.9999 to 1.00000 means an extra clock
          between steps.
          if (fClocksPerStepCounter >= 10000) {   // Fraction is
                                      // scaled by 10,000 truncate
            fClocksPerStepCounter -= 10000; //  the overflow.
        iClocksPerStepCounter++;        // and add in
                                        // that extra clock.
        // Check if it's time to decelerate yet
      if (StepCount > Motor1Accel) {
        // No.  Not yet so we're accelerating.
        if (Motor1AccelReload > 0)      // Reload value set at
                                        // beginning of move.
          --Motor1AccelReload;
      }
      else          // Last few steps so let's decelerate by
        Motor1AccelReload++;    // slowly increasing the time
                                // between each step.
    // Add in accel/decel values
    iClocksPerStepCounter += Motor1AccelReload;
            bM1StepMotor = 0;   // Finish step pulse.
    StepCount--;                // Another Step Done
    // Update DRO.
    if (bM1DirectionMotor) MotorPosition++; else MotorPosition--;
    }
  }
  else if (fMoveBSY) {          // Move complete
    fMoveBSY = 0;               // Inform system move is complete
     INTCONbits.GIEL = 1;
  }
}
if (INTCONbits.INTOE && INTCONbits.INTOF) { // Spindle Encoder
     Interrupt
    INTCONbits.INTOF = 0;
    // Save # of CCP1 Pulse Clocks Since last Index.
    SpindleClocksPerRevolution = SpindleClockValue;
    // Start counting anew
    SpindleClockValue = 0;
    // Let Device Driver know there's a new RPM.
    fUpdatedRPM = 1;
    if ( fMoveRQ ) {// Always start threading at index pulse.
fMoveRQ = 0;
      fMoveBSY = 1;             // Ack Request
      Motor1AccelReload = Motor1Accel;  // Set up to accelerate
      iClocksPerStepCounter = iClocksPerStep;
      fClocksPerStepCounter = fClocksPerStep;
    }
  }
  bSTATUS = 0;  // Scope this bit to see how long interrupt takes.
}
```

face seemed like a waste of time, but I did accomplish what I set out to do: I cut threads with an electronic gearbox. The next step is to create a system that uses the algorithms from this project and has a useful user interface and stepper motor driver all rolled into one stand-alone package.

Author's Note: I'd like to thank Dave Hylands for his suggestions concerning the Bresenham algorithm and Bob Warfield for the spreadsheet I used for the initial calculations.

John Dammeyer holds a B.Sc. in computer science and electrical engineering from the University of Alberta. He designs and writes software for embedded systems and CAN bus systems. Green sand casting aluminum and bronze bits (with a computer-controlled foundry) for his sailboat and amateur machine shop is one of his many hobbies. You may contact John at johnd@autoartisans.com.

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2006/196.

RESOURCES

E-Leadscrew Group,http://groups.yahoo.com/group/E-Leadscrew.

D. Gingery, *Build a Metalworking Shop from Scrap*, David J. Gingery Publishing, Lindsey Publications, www.lindsaybks.com.

D. Ramirez, "Math Coprocessor for Robotics Applications," *Circuit Cellar* 172, 2004.

SOURCES

MACH 2 CNC Software
ArtSoft Software
www.artofcnc.ca

Microstep driver
Embedded Acquisition Systems
www.embeddedtronics.com

MCC18 Compiler and PIC18F2620 microcontroller
Microchip Technology, Inc.
www.microchip.com

Low-Cost Gas Leak Detector

A gas leak in the chemistry lab in which Brian works can mean big trouble for the electronic instruments he uses on a daily basis. That's why he built an ATmega88-based gas leak detector. The system enables him to monitor for small leaks around the lab's expensive equipment.

Depending on what you do for a living, the terms leak and leakage can mean different things, but most of us would agree that it's never a good thing. To electronics professionals, it means a degradation of circuit function at best and a safety issue at worst. And while it might put food on the table for a plumber or an auto mechanic, water and oil leaks are a nuisance that most of us can do without. Even computer programmers must cope with memory leaks, those insidious programming bugs that eat up more and more memory over time as a program runs.

I deal with a broad range of scientific instrumentation in the chemistry lab at Dalhousie University where I work. Most scientific analytical instrumentation involves various forms of gas or liquid flows and high vacuum requirements. Therefore, I sometimes get sidetracked from my normal electronics duties when instruments fail to operate properly due to various problems involving liquid and gas flows. Although I have excellent diagnostic tools for electronics troubleshooting, I've never worked steadily enough on liquid/gas flow or high vacuum systems to justify buying a good leak detector. Partially, this is due to the cost of such systems, but it also reflects the reality that there are many different instruments available to handle the many types of possible leaks.

In this article, I'll describe a homebrew gas leak detector that works by pumping up your system with helium gas to a pressure slightly above atmospheric, and then using the detector's probe to "sniff" around for locations where the helium is leaking out. I take no credit for the detection method used in this leak detector: the concept has been used in gas chromatography since before I was born and adapted to portable leak detectors more recently. However, you can build my leak detector for less than $100, whereas commercial units can cost more than $1,000.

DIFFERENTIATING GASES

Like liquids and solids, gases possess the ability to conduct heat. The measure of this property is called a substance's thermal conductivity. All gases don't conduct heat equally. As you can see in Table 1, the higher the number, the better that gas is at conducting heat. You can see that gases such as hydrogen and helium have thermal conductivities six to seven times greater than air. This makes such gases easy to detect even in the presence of our normal atmosphere. In contrast, air is made up of 85% nitrogen and a bit less than 15% oxygen, so it's no surprise that the thermal conductivity of air is very close to that of nitrogen. This rules out nitrogen as a good gas for use in thermal conductivity leak detection methods.

Gas	Thermal conductivity (W/mK)
Hydrogen	0.1805
Helium	0.1513
Oxygen	0.0266
Nitrogen	0.0258
Air	0.025
Argon	0.0177

Table 1—*Made up of mostly nitrogen, air has a thermal conductivity that is very close to it. In contrast, helium and hydrogen have much higher conductivities.*

If you assume that it's possible to measure a gas's thermal conductivity easily, then it follows that a good way to detect leaks would be to pressurize a leaky system with either hydrogen or helium and then physically sniff around the system's surface to look for the presence of this introduced gas. Because of the large difference in thermal conductivity between either of these two gases and air, even small amounts of these gases should be detectible given the overwhelming abundance of air surrounding the system. The fact that both helium and hydrogen have very low molecular weights means that they will leak through small orifices and cracks very easily, which also improves the viability of this method.

Although hydrogen has a slightly higher thermal conductivity than helium (7.22 versus 6 relative to air), it is highly flammable and therefore not too desirable in most cases. There are leak detectors that use hydrogen as the detected gas, but they use a gas mixture containing only about 15% hydrogen, which addresses the flammability issue. This gas mixture does not work well with thermal conductivity detectors (TCDs), so more esoteric detection methods must be used. Although helium is not generally readily available to the general public (except in balloons!), it is readily available in many science labs.

TCDs EXPLAINED

As I already mentioned, thermal conductivity detectors have been

around for a long time. They even predate their common use in gas chromatography. They work by heating a wire filament to a temperature significantly higher than the surrounding gas. The surrounding gas conducts away some of the filament's heat. The amount of this heat loss is proportional to the thermal conductivity of the gas. This heat loss results in the lowering of the filament's temperature. Because the resistance of a metal varies with temperature, you can measure the heat loss by somehow measuring the wire's resistance while it is simultaneously passing the heating current. These resistance changes are quite small, so it was common to configure such detectors using a Wheatstone bridge (see Figure 1).

All four filaments are identical: two diagonally opposed filaments are placed in a reference gas, and the other two filaments are placed in the gas whose thermal conductivity is being measured. The filament bridge is excited by a stable DC voltage capable of providing enough power to heat all four filaments up to a temperature significantly higher than that of its surroundings. When a difference in the thermal conductivity of the sample gas versus the reference gas exists, the temperature of the two sample filaments will differ from that of the two reference filaments, causing a resistance imbalance. This will unbalance the bridge, the output of which can be amplified and measured with a voltmeter, chart recorder, and so on.

The TCDs used in gas chromatography are basically implemented like this. Wire filaments are used because they must be heated to relatively high temperatures (necessary in gas chromatography work), even though the change in resistance with temperature of most metals is very low, reducing the sensitivity of the system. However, for the purposes of leak detection, the temperature of the gases can generally be in the normal room temperature range. This means that the heating filaments do not need to get too hot. For example, if you can settle for just heating the filament to something less than 120°C, you can dispense with using wire filaments and substitute thermistors. While thermistor properties vary greatly depending on chemical composition, the ones that are most suitable for this purpose are those that exhibit a negative temperature coefficient (NTC).

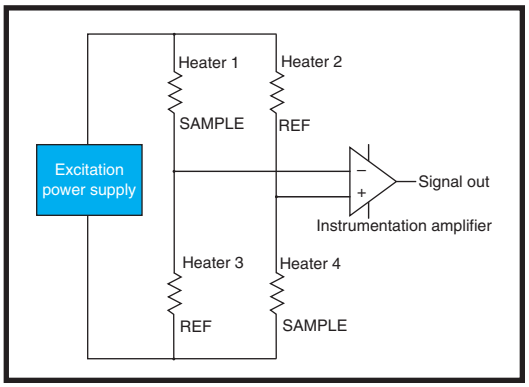


Figure 1—*A classic thermal conductivity detector is built using a Wheatstone bridge.*

NTC thermistors exhibit a drastic reduction in resistance with even moderate temperature changes, so they are much more sensitive to changes in temperature than the wire filaments used in the gas chromatography TCDs. Because of this, they work very well in leak detectors. As you can see in Figure 2, I plotted the response of the thermistor that I used in my leak detector. As you can see, its 10-kΩ resistance at room temperature drops dramatically to about 1,300 Ω at 80°C.

If you connect an NTC thermistor up to a high enough regulated voltage, it will heat up. The power it dissipates is defined as follows:

$$\text{Thermistor-dissipated power} = \frac{(\text{thermistor voltage})^2}{\text{thermistor resistance}} \quad [1]$$

As the thermistor heats up, its resistance drops rapidly, causing the right-hand denominator in Equation 1 to decrease, further increasing the power dissipation. A quick analysis of this relationship shows it to be the equivalent of positive feedback in an amplifier. Such a configuration will result in the eventual burnout of the thermistor if it is connected to a power supply of sufficient current capacity. You don't want that.

Contrast this with a configuration in which you feed the thermistor using a constant current source. Again, assuming you use a high enough constant-current, the thermistor will heat up. Now its power dissipation will be:

$$\text{Thermistor-dissipated power} = (\text{thermistor current})^2 \times \text{thermistor resistance} \quad [2]$$

Now, as the thermistor heats up and its resistance decreases, its power dissipation decreases because the thermistor current is fixed. This is equivalent to a negative feedback loop, and it quickly results in an equilibrium condition, where the heat produced by the thermistor remains constant. In this state, the voltage across the thermistor also remains constant, unless there is some change in the amount of heat that is being conducted away from the thermistor. This will happen when gases of varying thermal conductivity flow past the thermistor, making this configuration a suitable thermal conductivity detector.

The effectiveness of the negative feedback relationship in this circuit can be readily observed. In my detector, when varying the excitation current between 3 and 6 mA, the thermistor's resistance, which reflects its temperature, will vary to a such a degree that the voltage across the thermistor changes very little once the thermistor has had enough time to reach its new thermal equilibrium.

The end result is that one can replace the four-wire filaments, config-

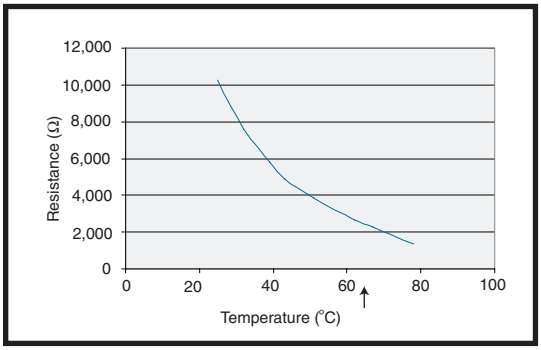


Figure 2—*In my leak detector, I run the thermistor in a self-heated mode at about 65°C.*

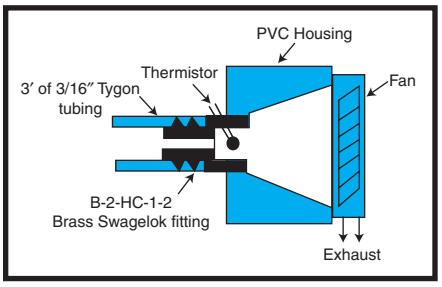


Figure 3—*Check out the home-brew pump and sensor assembly.*

ured in a Wheatstone bridge as used in gas chromatography, with a single NTC thermistor fed by a constant-current power source. This concept is commonly referred to as running a thermistor in Self-Heated mode.

LEAK-TESTING TCD

For sensitive leak detection, the amounts of helium gas leaking out will be quite small. I use 3′ of 3/16″ Tygon tubing to conduct the gases into the unit's thermal conductivity detector. The gas flows through the 0.125″ inner diameter of this tubing, making it necessary to use a thermistor which is smaller than this size, so that the gas flowing out of the end of this tubing completely envelops the thermistor as it passes by it. In this way, the thermistor exhibits the maximum possible change in resistance for a given volume of gas flowing by it. Similarly, given the small volumes of gases involved, it requires only a small vacuum pump to draw the gases into the Tygon tubing for subsequent detection. Indeed, if a larger pump were used, the resulting high gas flow rate would move the gases by the thermistor so rapidly that it would be unable to thermally respond quickly enough.

Small vacuum pumps capable of handling gases are a bit hard to find and tend to be relatively expensive. Therefore, I designed my own. My scrounging nature took over here. I used an ADDA AB0305HB-GA6 fan that I had on hand (mounted on an unused fan module from Cypress Semiconductor's Express development kit). Designed to cool laptop CPU chips and running on 5 V, the fan is ideal for my battery-powered device. With all of the cast-off laptops around, I'm sure there are lots of these to be had. Figure 3 is a cut-away drawing showing the vacuum pump (fan) and the thermistor sensing assembly. Photo 1 shows the assembly. The thermistor wires are exiting the Swagelok fitting that I used as part of the assembly.

I looked high and low for a tiny bead-type NTC thermistor. I wanted one that had a bead diameter less than 1.5 mm so that the gas flow would completely envelop it and give the best sensitivity. Google has spoiled me. I expect to be able to find just about anything I need online. Even if it's not available in small quantities, I've had great luck obtaining samples from manufacturers. In this case, though, I struck out until I tore apart an Omega TPD38 hand-held digital thermometer. The thermistor in this unit is a 10-kΩ NTC bead-type with a diameter of only 1.2 mm, which is the perfect size for this application. We use a lot of these digital thermometers in our teaching labs. They cost about $20, so we don't bother trying to fix them when they fail, and I had a couple of the broken units on the bench. I determined that the thermistors themselves are pretty well matched from one unit to the next.

EXPERIMENTAL STEPS

Before I built the electronics circuitry, I fabricated a prototype of the fan/sensor assembly so that I could determine the operating conditions that would be needed to detect small amounts of helium gas. By experimentation, I found that I needed about 6 mA of thermistor current (from a constant-current source) to heat it to about 65°C, with the airflow provided by the fan running at 5 V. At this self-heating temperature, the thermistor was quite sensitive to even small amounts of helium gas introduced at the other end of the 3′ Tygon tubing. At this early stage, I just used a digital multimeter to measure the voltage across the thermistor itself. I borrowed a commercial leak detector unit for sensitivity comparisons, and my detector compared favorably with the commercial unit.

The amount of suction provided by my fan-type vacuum pump was enough to draw in the gas reliably. It took less than a second for a small puff of helium gas released near the probe end of the tubing to propagate its way into the detector and show up as a voltage change across the thermistor. I measured the suction draw of the pump by immersing the probe end of the Tygon tubing in water and observing that the fan would draw about 0.375″ of water vertically, corresponding to a negative pressure of about 0.015 psi! This seems tiny, but it is sufficient for the job at hand.

As you can see in Figure 2, given the 6-mA current necessary to achieve thermistor self-heating to about 65°C, there will be about 13.5 V across the thermistor in the presence of airflow (6 mA × 2,250 Ω). When helium flows past the thermistor, the helium's much higher thermal conductivity cools the thermistor down. The cooler thermistor exhibits a higher resistance, and because the current is a constant 6 mA, the voltage across it rises.

It made sense to tie one end of the thermistor to an adjustable negative voltage source of about –12 V. The other end is connected to a 6-mA constant-current source (provided by a LM334 constant-current device driven from the 5-V power supply). In this configuration, if you measure the voltage at the latter thermistor terminal, it will vary from 1 to 2 V with airflow up to a voltage of about 4 V when helium is present in strong concentrations. This is a good range given the full-scale range present in the ADCs of most microcontrollers. With the success of this experiment, the next step was to design the leak detector's electronics.

BUILDING THE UNIT

I decided early on to attempt to make the leak detector a small, battery-operated unit, because that is how commercial units are designed. I knew the unit would be used only sporadically and for short periods of time, so I decided that a battery pack of four alkaline AA cells would likely provide enough power (and the shelf life on these cells is several years). I estimated that I would need about 250 mA for everything because the fan itself requires

about 130 mA. At this rate, the battery pack should last a couple of hours.

The complete unit is shown in Figure 4 (p. 48). Starting with the power supply, I use the 6-V battery pack to directly supply the fan and the Texas Instruments DCP05012BP 12-V isolated power module. The DCP05012BP is highly efficient (85%) and provides about –14 V when loaded at about 10 mA, which is all the current I need. The negative supply is regulated by an MC79L12 with its ground terminal referenced to a potential controlled by a potentiometer. This allows the negative supply to be adjusted in the –7- to –12-V range. This adjustment will be discussed later. Because the LCD and microcontroller work best at 5 V, I elected to drop the 6-V battery voltage down to 5 V using a 100-Ω resistor and a 5-V Zener diode. If I had had a switching-mode regulator on hand, I could have saved a bit of power by using that instead.

The constant-current source for the thermistor is made up of an LM334 along with current set potentiometer R1 and R2. The LM334's current is set by the following relationship:

$$Current = \frac{64\ mV}{R1 + R2} \quad [3]$$

To get a 6-mA current, you need a total resistance of 10.66 Ω.

I used two current-setting potentiometers and a switch to handle both a low and a high sensitivity range. With the LM334 set to produce a 6-mA current, the thermistor will heat up and settle to a temperature of about 65°C, with the fan running and air passing over it. This will result in about 13.5 V across the thermistor. You adjust R23, the negative power supply adjust potentiometer, to bring the thermistor's output voltage (at the LM334 end) to about 1.5 V.

The thermistor voltage is fed to the noninverting input of an instrumentation amplifier consisting of three sections of a TLC2254 quad low-power, rail-to-rail op-amp. The main function of this amplifier is to shift the quiescent thermistor output voltage down to around 0 V. This is accomplished by feeding the appropriate offset voltage into the inverting input of the instrumentation amplifier. This offset voltage is generated by a microcontroller-based PWM DAC, which has a range of 0 to 5 V. Therefore, it's important that the negative power supply be adjustable to bring the thermistor's output voltage up to about a positive potential in

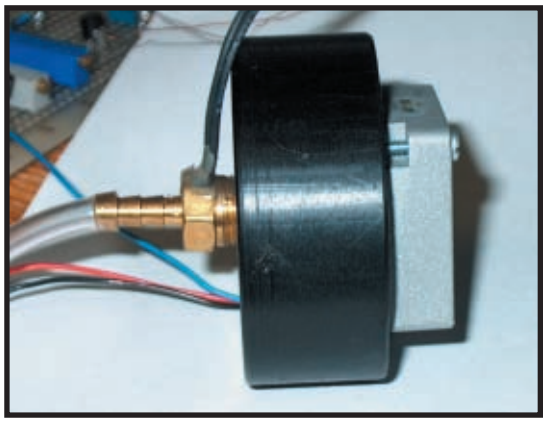


Photo 1—*The home-brew thermal conductivity detector consists of a thermistor mounted within a Swagelok fitting, a custom plastic chamber (black), and a small laptop CPU cooling fan.*



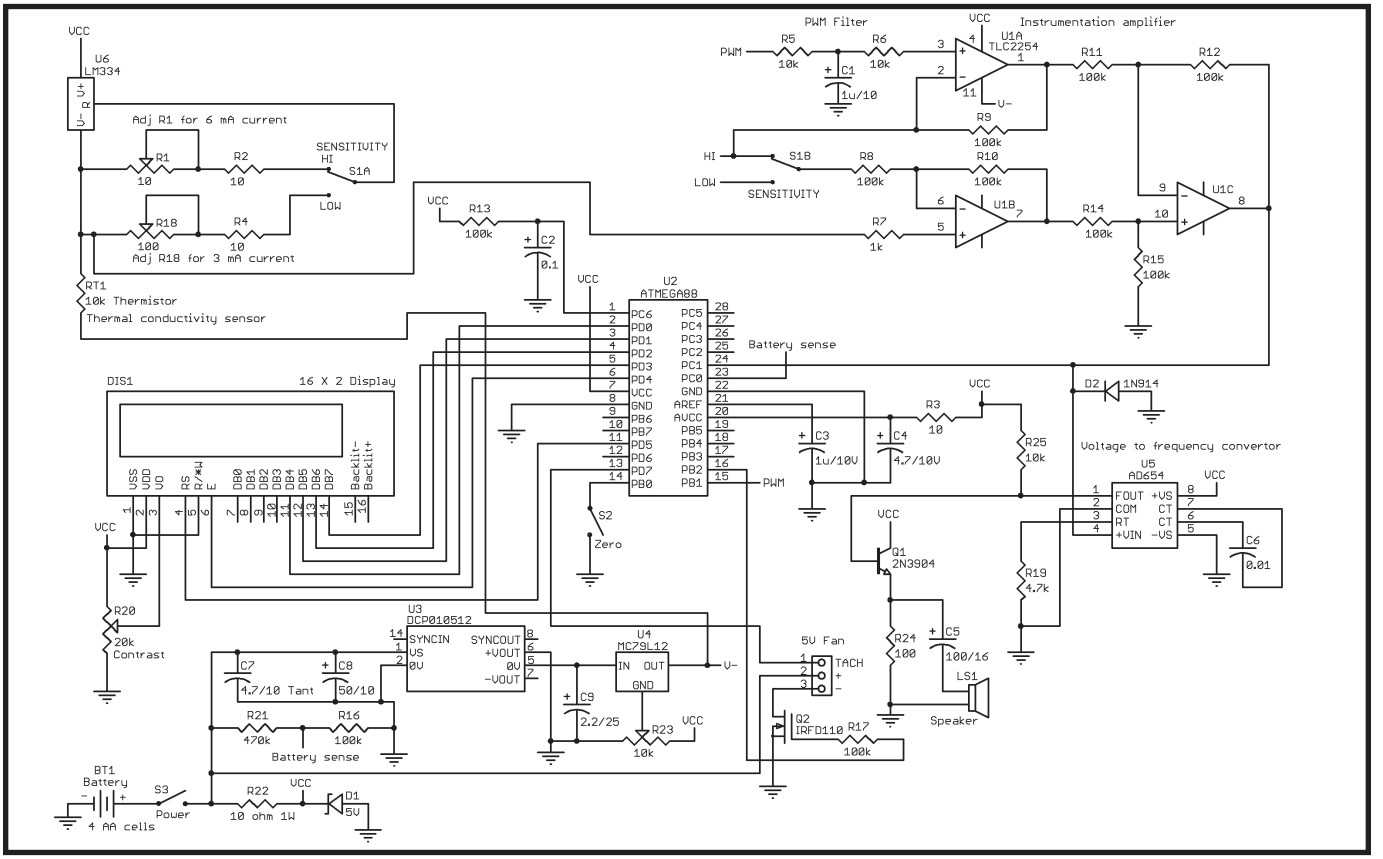


Figure 4—*Study the wiring closely. The thermal conductivity detector is made up of the components in the upper left of the diagram, as well as the fan, which you can see on the bottom right.*

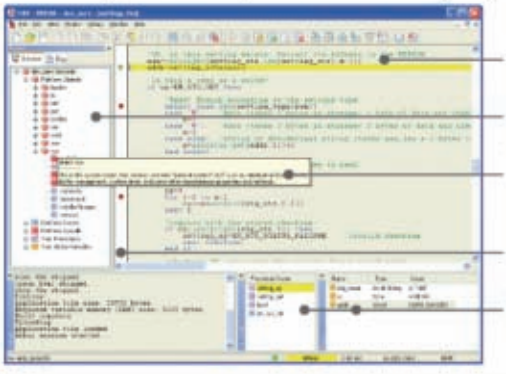
Tibbo TECHNOLOGY

Build your next automation project around our EM1000 BASIC-programmable Embedded Module



- 50 MIPS CPU
- 100BaseT Ethernet port
- 512K flash disk
- 4x high-speed UARTs
- High-speed parallel slave port
- Real-time clock with backup power
- 49x general-purpose I/O lines
- Development kit available (EM1000-SK)

Code and debug your Tibbo BASIC application using Tibbo Integrated Development Environment (TIDE) software



- Programmable – in BASIC!
- Optimized for real-time applications
- Rich object set
- Built-in webserver
- Event-driven operation
- Sophisticated development environment supports cross-debugging (no ICE needed)

web: www.tibbo.com email: sales@tibbo.com

the 1 to 2 V range, allowing this amplifier to work properly. The value of R8 sets the gain of the instrumentation amplifier. In practice, I found that a gain of one was adequate for the low sensitivity range. This gain requires no resistor to be mounted. To achieve higher gains, the following formula is used:

$$Gain = 1 + \frac{200\ k\Omega}{R8} \quad [4]$$

I settled on a gain of three for the high sensitivity range, so R8 = 100 kΩ.

The instrumentation amplifier's output connects to two circuits. One is the PC1/ADC1 pin on the Atmel ATmega88 microcontroller. The ATmega88's 10-bit ADC in this case is set to use the internal 1.1-V reference, resulting in a full-scale reading of 1.1 V. At start-up, the ATmega68's firmware goes through a routine that adjusts the PWM duty cycle at pin PB1/OC1A. This PWM waveform, when filtered by R5 and C1, provides the equivalent of a 12-bit DAC with an output voltage range of 0 to 5 V. This voltage is used as an offset voltage to the instrumentation amplifier. At start-up, the firmware adjusts the PWM DAC to bring the output of the instrumentation amplifier to about 0.1 V. When helium is present, this voltage will rise, and the ADC will be able to measure and display it.

If there was a shortcoming in the

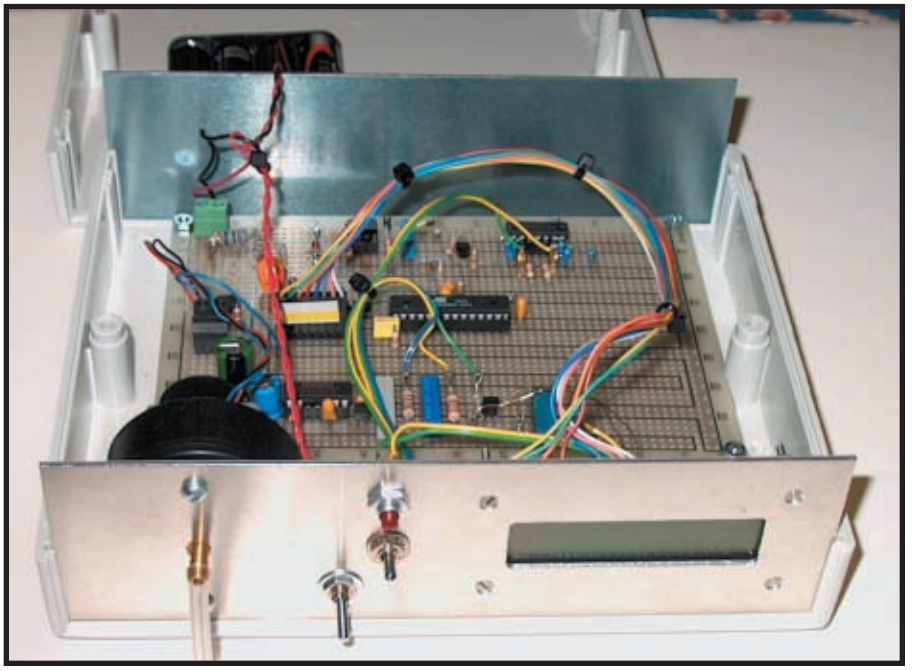


Photo 2—*Check out the complete unit with the top cover off. The battery pack is mounted in the top cover and is barely visible at the back.*





RoboDevelopment
Conference and Exposition

Producedby

roboticstrends

The International Technical Design and Development Event for Mobile Robotics and Intelligent Systems Industry



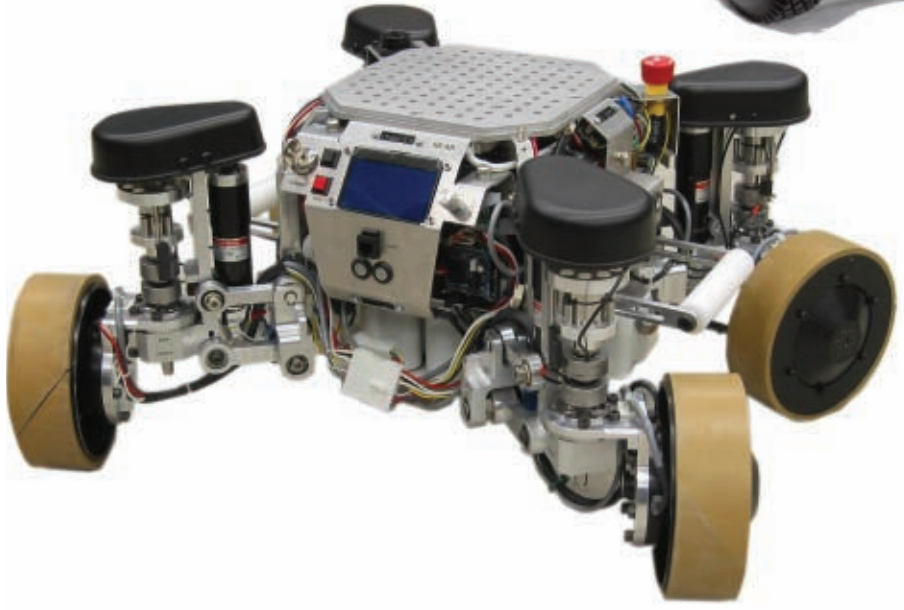
December 12-13, 2006
Santa Clara Convention Center
Santa Clara, California

Tracks include:

- Design, Development and Standards
- Tools and Platforms
- Enabling Technology



www.robodevelopment.com



Founding Sponsor

Premier Media Sponsor

Premier Association Sponsor

Corporate Sponsors

Media Co-Sponsors

Association Co-Sponsors

commercial leak detector I borrowed, it was that it has only a small 16-segment LED bar graph readout display. This forces the operator to look at the instrument itself while at the same time trying to poke the probe around the various nooks and crannies of the system, sniffing for the leak. This is somewhat inconvenient. I wanted an audible readout in addition to the LCD bar graph readout that the microcontroller was going to supply. A circuit that provides a tone whose frequency varies with the thermistor voltage is ideal, because the human ear is very sensitive to even small changes in frequency. Anyone who has installed and aligned a TV satellite dish will be familiar with this audible-feedback tuning process.

Therefore, the instrumentation amplifier also drives the input pin of an Analog Devices AD654 voltage-to-frequency converter chip. With the value of the components chosen, this VFC produces a square wave of a maximum frequency of 10 kHz when fed with a 5-V input. Because the microcontroller's ADC has a full-scale range of only 1.1 V, the audible feedback circuit covers a much wider range. In practice, the ear is most sensitive to frequencies below 2,500 Hz. With a full scale reading on the LCD, the tone is about 2,500 Hz, but higher helium concentrations still generate tones up to about 10 kHz. Because I wasn't too concerned with high fidelity here, I used only Q1, an emitter-follower, and a decoupling capacitor to provide enough drive current for a small PCB-mount speaker.

A 16 × 2 LCD is used for visual feedback. At start-up, the instrument first checks its battery and provides a bar graph indicating battery voltage as well as a Battery Good/Bad text message. Because the fan contains its own tachometer, I route that tachometer signal to the ATmega88's analog comparator pin. The 2-V_{PP} tachometer signal is compared to the 1.1-V reference internally, producing a clean pulse signal, which is used as an interrupt. This ISR counts tachometer pulses for a few seconds at start-up, and the fan's health is checked and reported on the LCD. The final step during start-up is to wait until the thermistor has had time to heat up and then incrementally adjust the PWM DAC until the instrumentation amplifier output settles at about 0.1 V. This is the desired state with the fan drawing air over the thermistor and an absence of any helium gas in the mixture.

During operation, as the probe sniffs the system, any leaks show up as a rise in pitch of the emitted tone, as well as a longer bar on the LCD bar graph. An 80-segment bar graph is provided in the firmware that uses the 16 characters across the screen (further subdivided into five smaller increments, using five custom LCD characters). During operation, if the thermistor sensor drifts by any great amount, a Zero button is provided, which, when pressed, will repeat the zeroing operation that was performed at start-up. This Zero function is a switch attached to the ATmega88's Port B.0, which is polled in the firmware. I initially mounted a MOSFET switch to control the fan speed under PWM control from the ATmega88. In the end, however, I settled on basically running the fan full tilt, so this circuit could have been eliminated.

The entire circuit, including the quad-AA battery pack, fits easily into a Hammond 1598D plastic cabinet, which measures about 8″ × 7″ × 2.5″ (see Photo 2, p. 49).

FIRMWARE

The firmware is straightforward. I wrote it using the Bascom-AVR BASIC compiler (V.1.11.8.3). The source and object code for the project are available on the *Circuit Cellar* FTP site.

The ATmega88 that I used is in the middle of a family of AVR microcontrollers comprising the ATmega48, ATmega88, and ATmega168 models with 4, 8, and 16 KB of flash program memory, respectively. This application uses only 2,200 bytes of flash memory, so it's possible to substitute a 4-KB ATmega48 without any other changes. Because the ATmega48 is only about $1.50 less expensive, I don't bother stocking it. I use the pin-compatible ATmega88 and ATmega168 instead. Because there weren't any timing-critical aspects to this application, I chose the internal 8-MHz RC clock option for the ATmega88 and saved a crystal.

SIMPLE SOLUTION

I am pleased with my detector's operation because I don't have to borrow a commercial unit whenever I need to check for small gas leaks. As I mentioned, I scrounged the fan and thermistor, and the rest of the components were just common parts I keep on the bench.

I have to confess that once I had an experimental version of the detector built and working, it was a toss-up deciding which microcontroller to use to complete the electronics portion of the circuit. Because I'm most familiar with Atmel AVR MCUs and the Bascom-AVR compiler, I finally settled on the ATmega88. There was an Atmel contest underway when this project cropped up, and that factored into the decision too. However, I think I could have squeezed almost all of this circuit, including most of the analog functions, into a Cypress Semiconductor CY8C29466 PSoC. I've done some recent projects with the Cypress PSoC devices and they have been very impressive in mixed-signal applications.

Brian Millier (brian.millier@dal.ca) is an instrumentation engineer in the chemistry department at Dalhousie University in Halifax, Canada. He also runs Computer Interface Consultants.

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2006/196.

SOURCES

AD654 Converter
Analog Devices, Inc.
www.analog.com

ATmega88 Microcontroller
Atmel, Inc.
www.atmel.com

TPD38 Digital thermometer
Omega Engineering, Inc.
www.omega.com

DCP010512BP Power module
Texas Instruments, Inc.
www.ti.com

TV-Based Oscilloscope

Ilya turned an ordinary TV into a digital storage oscilloscope with a sampling rate of 160 ksps. In this article, he explains how to use an LPC2138 microcontroller to build a high-resolution system of your own.

There are many low-cost solutions (e.g., sound card-based) based on USB stand-alone devices that will turn your PC into a digital storage oscilloscope (DSO). All of them have a significant disadvantage: when you use the PC for a measurement and other functions at the same time, you must click your mouse again and again to switch between applications. Of course, you may use a dedicated notebook, but the other disadvantages will remain.

You could also use your old TV with a special add-on tool. Advanced microcontrollers have all the parts (e.g., ADC and RAM) for implementing such a tool with a single chip. I'll describe a similar device in this article. My TV-based oscilloscope is built around a Keil MCB2130 evaluation board, which you can easily turn into a DSO with few passive components. Note that this project requires a code size of only 16 KB, so you can use the evaluation version of the Keil CARM compiler.

Implementing the oscilloscope was not my main task. I usually try new microcontrollers in video applications, so I figured this would be a good way to demonstrate the capabilities of the NXP (founded by Philips) LPC2138 microcontroller for generating a high-resolution (512 × 240 pixels) video picture.

SIMPLE VIDEO SIGNAL

A standard video signal is a sequence of sync pulses and the analog waveform between them. When there are two gradations of brightness, the signal has three levels (see Figure 1). There are two basic monochrome standards—RS-170A and CCIR (typically called NTSC and PAL for their color varieties)—that have different timings, lines per frame, and refresh rates.

Note that the standard video signal has a complex structure of sync pulses. Each video frame has two nonidentical fields (odd and even) with a fractional number of TV lines for each. Sometimes the structure can be simplified (e.g., in the case of the identical odd and even fields). Normally, the TV accepts the nonstandard video format without glitches.

As for generating the signal, the microcontroller must be programmed to output both the video bits and the sync pulses. This task was traditionally solved by software, but such a technique doesn't produce enough elements in a TV line. Therefore, you must integrate the other tasks into the video-generating algorithm and calculate all of the durations to avoid the probable artifacts and the synchronization glitches.

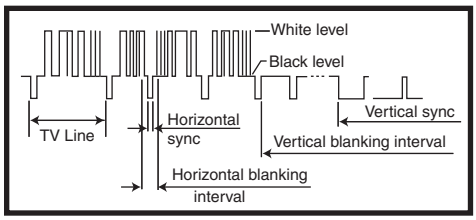


Figure 1—*This is the structure of a monochrome video signal. This project uses the simplified structure of the vertical sync pulses (without equalization pulses). The number of TV lines per frame is even.*

VIDEO WITH THE LPC2138

In order to obtain high-resolution video, I had to generate a TV signal via the hardware as much as possible. The LPC2138 microcontroller has a PWM block and SPI communication module. The former can be used for generating sync pulses. The latter can be used to generate the video stream! Want to know a secret? I already tried this with Cypress Semiconductor PSoC chips. Unfortunately, the capabilities of the PSoC SPI module did not allow me to obtain a horizontal resolution of more than 256 pixels.

The useful feature of the LPC2138's (also called the synchronous serial port (SSP)) SPI1 block is the fact that it can operate in a continuous SSI mode. It has an eight-frame FIFO buffer and can send 16-bit frames. This means that it can process a 128-bit block without participation from the software. Of course, to send more bits, the software must reload the FIFO buffer with additional data.

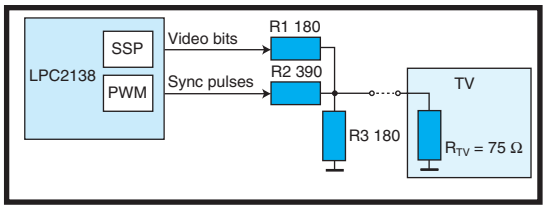


Figure 2—*SPP and PWM blocks generate video. The matrix of R1, R2, and R3 composes the external DAC.*

An external DAC is required to mix both the video and sync pulses. It can be built using a resistor matrix like the one shown in Figure 2. With a 3.3-V supply and a 75-Ω load (a TV), this matrix produces the sync and white levels of approximately 0.3 and 1 V. The output impedance of this simple DAC is approximately 75 Ω

for compatibility with 75-Ω coaxial cable. In reality, the level of black is a bit shifted up after the last and before the first video bit inside the TV line because the SSP block has a high output impedance in Idle mode.

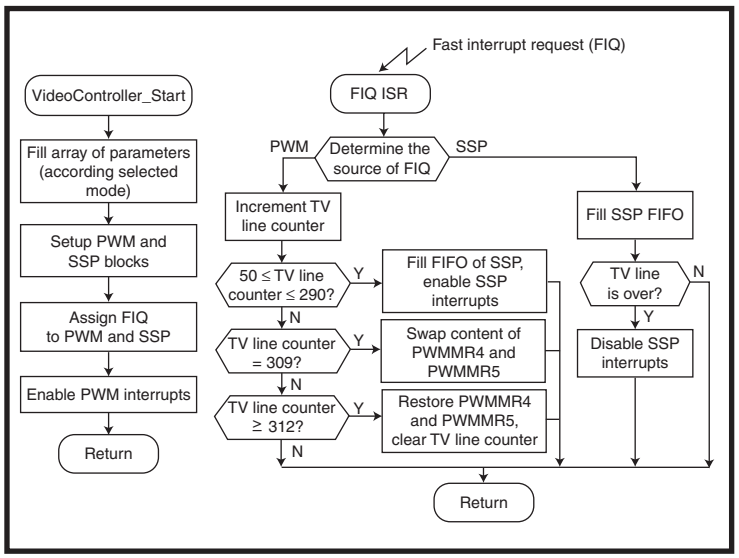


Figure 3—*The specific values for comparisons correspond to the PAL mode. NTSC uses different values.*

VIDEO ALGORITHM

The algorithm for generating a single TV line is simple. Every 64 µs for PAL (or 63.536 µs for NTSC), the PWM block produces a horizontal sync pulse and simultaneously generates an interrupt that initiates the sequence of SSP cycles for a single TV line.

Because the SSP module has only 128 bits of its own memory (eight 16-bit FIFO frames), it can't contain all the data for a 512-pixel line. To create a continuous bitstream, a new chunk of data should be loaded into the SSP as soon as there's room for it. For this purpose, I use the "half empty" interrupt. This means that every time the FIFO becomes half empty, it requires the new portion of data. I assigned both the PWM and SSP interrupt requests to the FIQ category because the FIQ has a high priority and the fastest possible latency (see Figure 3). The disadvantage is that the FIQ handler should first identify which source is requesting the interrupt. Although it isn't a recommended method, this is a unique way to create a glitch-free video when the other tasks use the vectored interrupt controller. (In this case, they must use only the IRQ category of interrupts!)

As you can see in Figure 3, when the FIQ is identified as coming from the PWM module, the SSP module's FIFO is completely filled and its interrupt is enabled. This means that the SSP starts to output the content bit by bit and frame by frame. When the FIQ is classified as coming from the SSP module, this means that the FIFO is half empty and the next data chunk should be loaded into the FIFO again. When all of the chunks for a single TV line have been processed, the SSP interrupt is disabled for the next interrupt from the PWM.

The SSP module has a specific prescaler so a required bit rate can be chosen from fixed values: b = 60 MHz/2*n* (i.e., b = 30, 15, 10, 7.5 Mbps, etc., where 60 MHz is the nominal clock of LPC2138). These values correspond to the horizontal resolutions of 1,536, 768, 512, 320 pixels, and so on. (The horizontal blanking interval is taken into account.) I achieved all of the resolutions in practice. Note that the maximum resolutions of 1,536 and 768 aren't applicable for color TVs because the color cathode ray tubes have discrete three-color dots on the surface of the screen.

To obtain the required TV frame structure, a TV line counter is provided in the software. The counter is incremented every time the PWM interrupt occurs. The current value of this counter is compared with a specific number to trigger a specific operation at a particular instant. Specific numbers define a vertical position, the height of the picture, and the duration of the vertical sync.

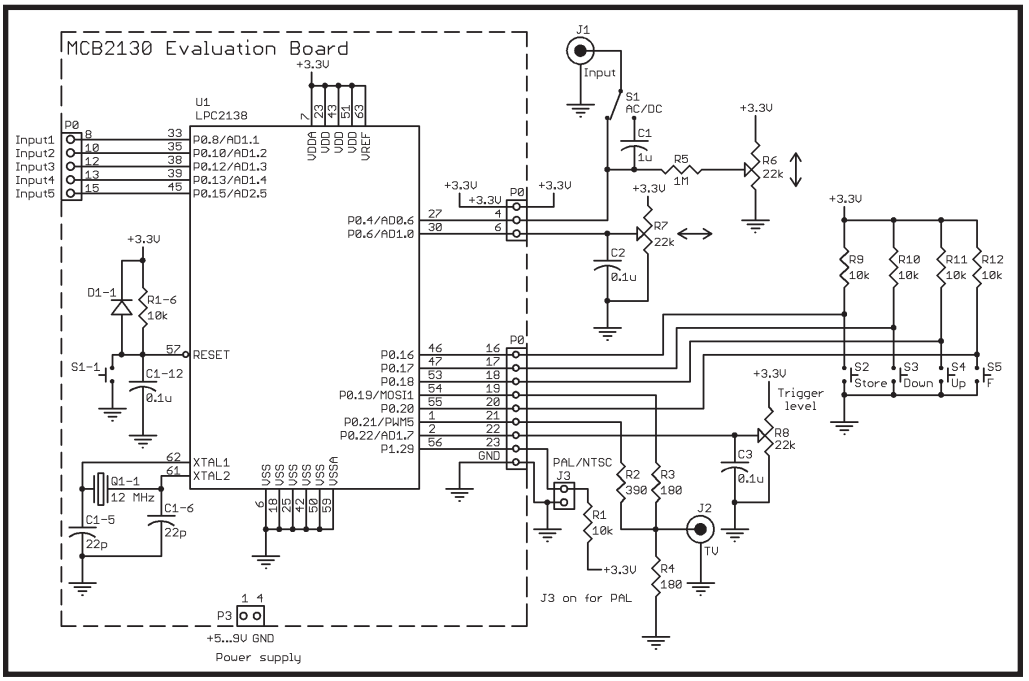


Figure 4—*The oscilloscope's layout is fairly simple. The Keil MCB2130 evaluation board is shown on the left. I didn't include unused components on the board.*

TRICKS USED

The SSP module begins to send the bitstream immedi-

ately after the first word is loaded into the FIFO. Any random deviations (caused by multitasking) of delay between the PWM interrupt request and the writing into the SSP data register will cause the jitter of horizontal lines.

To prevent this problem, I use a pair of PWM interrupts (instead of a single one) with the minimum possible duration between them. After the first request, all of the interrupts (except the PWM) are disabled and the CPU is forced into Idle mode. Next, the second interrupt from the PWM module resumes the CPU with a predictable delay.

VERTICAL SYNC & RESOLUTION

According to the RS-170A and CCIR standards, the vertical sync pulse is a sequence of inverted horizontal sync pulses. In my program, they are obtained by inverting the PWM output. At the end of the TV frame, the content of the PWM match registers (defining the rising/failing edge positions) is temporarily swapped.

In my project, the number of TV lines per frame is even, so equalization pulses aren't required. It seems all of the TV models admit such mockery.

The RS-170A standard provides 242 theoretical visible lines between the vertical sync pulses. The CCIR standard does 287. I decided to use the same vertical resolution (240 pixels) for both standards, although a value of 256 or higher is more preferable for CCIR.

Almost all of the TVs overextend the raster to use the all screen area. As a result, the upper and lower parts of a generated picture can be invisible in RS-170A because the only two blank lines (2 = 242 – 240) are reserved here. Try to adjust the vertical size if your TV has this option, or select PAL mode if you have a multisystem TV.

VIDEO FOR OTHER APPS

The video-generating routines are located in the VideoController.c file posted on the *Circuit Cellar* FTP site. The critical fragments are written in assembler. The `VideoController_Start(mode)` function starts the video stream for a specified TV standard. (The parameter mode should be 1 for CCIR in Europe and 2 for RS-170A in the U.S.) When developing your own system, simply add the file to your project and call the function at the initialization section of your application. Use the `VideoArray` global variable for access to bitmap data. Just remember that the bitmap array has a half-word organization and is in big Endian format.

I also wrote a collection of helpful graphic routines that include general-purpose routines (e.g., drawing lines, text, wallpaper, and bitmaps) and specific functions (e.g., drawing a grid and printing stylized digits). All of the functions have comment headers. The fonts are also present there.

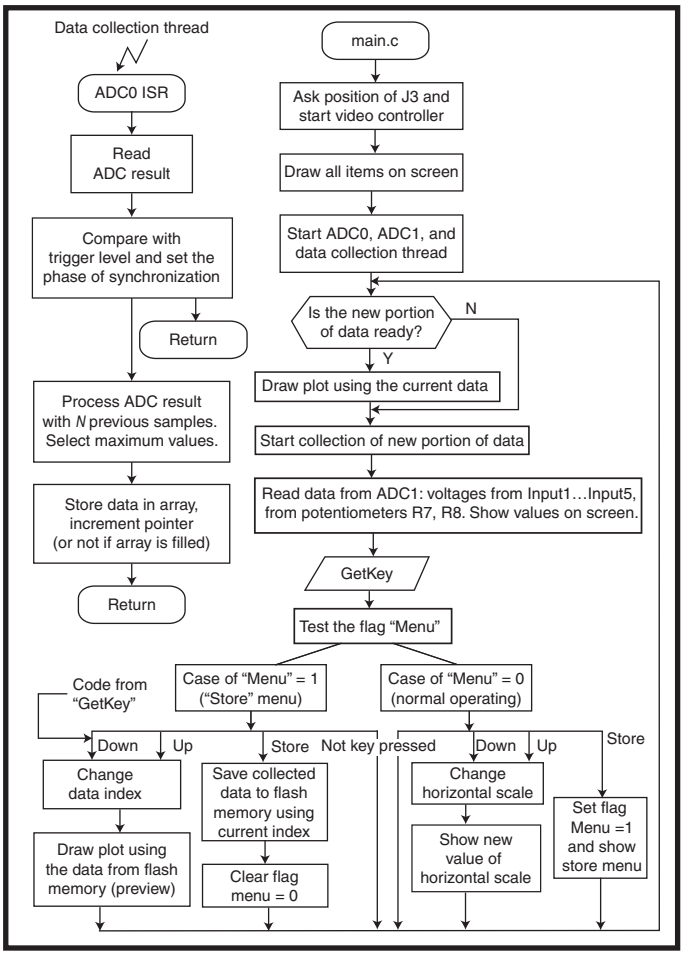


Figure 5—*The GetKey procedure doesn't suspend the loop; it just returns the code. Code = 0 when the keys aren't pressed.*

OSCILLOSCOPE

As I mentioned, the LPC2138 (IC1) is the heart of the system (see Figure 4, p. 53). I used the Keil MCB2130 evaluation board with a soldered LPC2138, a 3.3-V regulator, and in-system programming (ISP). The investigated signal comes in from the BNC connector (J1) to the internal ADC of IC1. The DC component can be removed from signal via the S1 switch. In this case, the potentiometer (R6) adjusts the vertical position of the graph.

Other components are used for control. R7 adjusts a horizontal position. R8 sets a trigger level (a voltage when the collection cycle should be started). S2 through S5 are control buttons used for setting the horizontal scale or calling the Store menu. J3 selects the European or U.S. video standard. (The position of jumper is read at power-on.) Resistors R2, R3, and R4 compose the video DAC. J2 is an output RCA connector.

The optional analog inputs (Input1 through Input5) can be used as voltmeters. For this purpose, the firmware continuously measures these inputs and displays the results on the TV screen. In this version, the range of input voltages must be from 0 to 3.3 V because the supply voltage (3.3 V) is used as a reference.

IMPLEMENTATION

The firmware consists of two parts, one for the video controller and one for the oscilloscope. All of the oscilloscope routines are in the Oscilloscope.c file posted on the *Circuit Cellar* FTP site. The main thread is in main.c file (see Figure 5).

The main loop, which

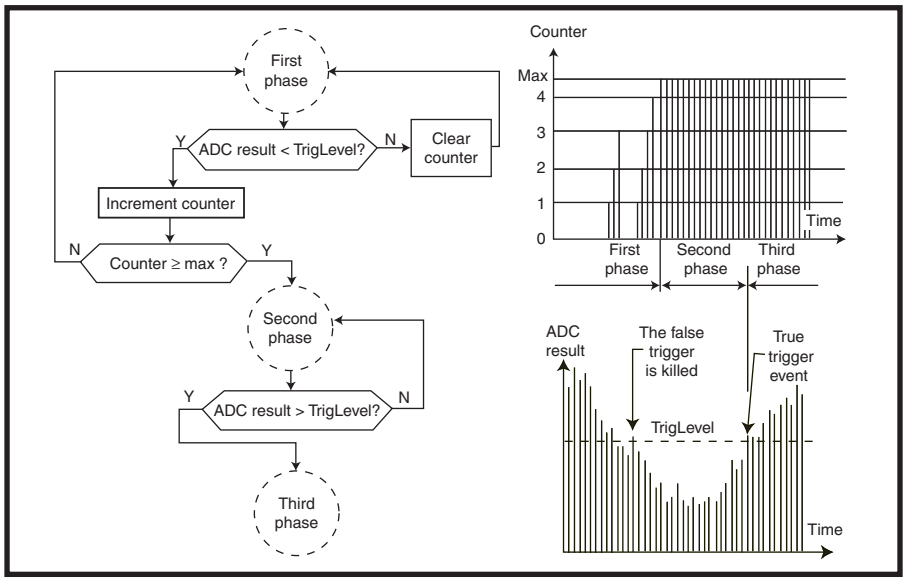


Figure 6—*This is the debouncing algorithm in the trigger mechanism.*

starts after initialization, manages data collection, visualization, and the device control. The special Menu flag assigns the current action for the Down and Up buttons. This flag is toggled with the Store button.

During normal operation (the Menu flag is 0), the Down and Up buttons make the timebase (horizontal scale) one step faster or slower. It's achieved by processing several ADC samples (the "N" in Figure 5) to obtain a single display sample, not by a variation of the ADC sample rate. This allows the system to implement Envelope mode in which the highest value and the lowest value are calculated for each group of *N* ADC samples and stored as the data of a single display sample. On the plot, a vertical line between the two values is drawn so glitches or other high-frequency components of the signal will always be visible. A line between two adjacent samples is also drawn.

The data collection thread fills the special array of samples and signals to the main loop that the data is ready. The main loop then waits in the TV frame to start the visualization of data. This is necessary to prevent any problems caused by interference between the video controller's refresh rate and the data collection rate. Four visualization algorithms are used for this purpose.

The first algorithm is for high data collection rates (i.e., horizontal scales). It waits until the data array is completely filled for one record view.

The second and third algorithms are for middle and low rates. They immediately visualize the current portion of data, which is collected between two adjacent TV frames. The difference between them is in a Trigger mode. At low collection rates, it isn't necessary to wait for the trigger event because there is enough time to consider the graph. As a result, the trigger system is off here.

The fourth algorithm (transient recorder) is intended for super-slow data collection rates. It immediately draws the new sample and also uses Roll mode. This means that the each new sample causes a change in the record view offset and makes the space for the new sample. For measuring slowly changing signals, it looks nicer than Scan mode, where the display area is filled on the left side when the graph reaches the right side.

The data collection thread is written as an ADC interrupt service routine (ISR) that's rich in functionality. It serves all of the visualization algorithms and uses the different conditions to start a data collection cycle. One condition is a trigger event that occurs when the signal crosses a threshold (trigger level). In this version, I detect the crossings of the rising parts of a curve. Comparing the current sample value and the trigger level value does this. The trigger value is calculated using the voltage from the



Add SD in a FLASH!

with our

Flash Card Library

- Easily exchange data between your product and a PC
- Create, Append and Delete PC compatible files
- Add GIGABYTEs of onboard or removable storage to your product
- Compatible with over 20 card styles SD, MMC, MicroSD, TransFlash, …
- No OS, RTOS or interrupts are needed
- Works with low cost Microcontrollers
- Support for FAT16 and FAT32 file systems
- Full C sources easily ports to many C compilers

Introductory Prices starting at $139.95

Flash Card Adapter Board

The flash card adapter board contains all the basic circuitry needed to add PC-compatible flash card memory to virtually any microcontroller application, using as few as four I/O pins. It also includes a 3.3V regulator, built-in level shifting, and sockets for both SD/MMC and MicroSD/TransFlash memory cards.



Introductory Price $39.00

PIC Development Board

The development board includes a built-in demo application and all the necessary hardware to run each of the example programs included with the SFCLIB (sold separately). It includes sockets for SD/MMC and MicroSD/TransFlash memory cards. The board also has an MPLAB ICD 2 In-Circuit Debugger jack for easy programming and debugging, and a large prototyping area for adding your own circuitry.



Introductory Price $99.00

EFFICIENT COMPUTER SYSTEMS, LLC

Offering MEGA solutions for the embedded MICRO world.

603-776-3151 • www.ECS87.com/sfclib

5 Emerson Drive Center Barnstead, NH 03225

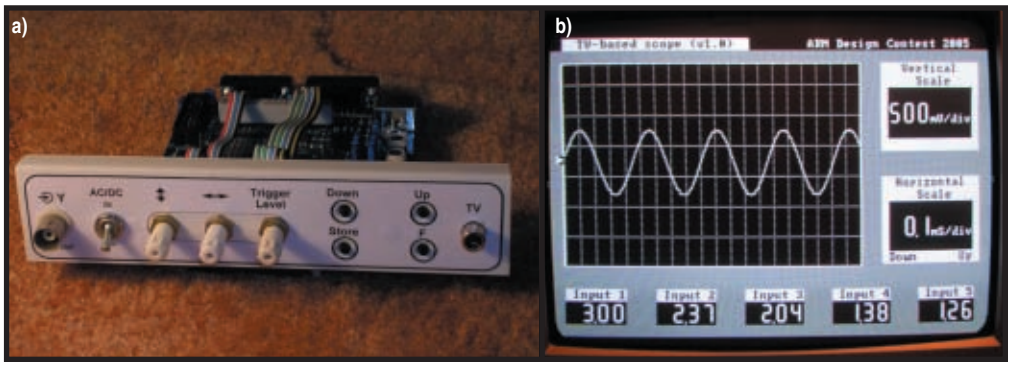


Photo 1a—*Ribbon cables connect the MCB2130 evaluation board to the front panel.* **b**—*The triangular cursor on the left side corresponds to the position of the Trigger Level knob on the front panel.*

rotating control located on the front panel.

The real signal has a noise component that causes false triggers near the cross points. The falling edge can be recognized as rising. To prevent this, I used a debouncing algorithm like the one commonly used for mechanical switches (see Figure 6, p. 55). Of course, this isn't the only solution. Many devices use variable hysteresis algorithms or HF filters.

The voltages from rotating controls also have a noise component, but another algorithm should be applied to kill it: math filtering. I implemented a simple procedure, moving average:

```
GetVoltages_TrigLevel=(GetVoltages
 _TrigLevel+val)>>1; //New (filtered)
 value of trigger level
```

Another function of the data collection thread is to create a delay between the trigger event and the start of data collection. This is equivalent to the beam offset function in analog oscilloscopes. The value for the delay is obtained like the value for a trigger level from the rotating control at the front panel.

The Store button stops the data collection process, captures the graph, and calls the Store menu. You can use the Down and Up buttons to move the special cursor along the row of numbers. The stored graphs are shown when you move the cursor to the required position. Each number corresponds to a stored graph that can be replaced by the current waveform when the Store button is repeatedly pressed. The Exit position causes the return to normal operation without saving anything.

Ample flash memory (512 KB in the LPC2138) enables you to implement a life-prolonging algorithm for a storage system. Normally, a single flash memory block has a limited number of erasure cycles. The data for one graph (1 KB) is considerably less than a block size (32 KB), so one flash memory block can contain a set of 32 (32 KB/1 KB) stored graphs. I use this technique when the flash memory

Fighting against your PCB-Design Software?

Here's something that will spare your time and your budget!

Boards designed under EAGLE are found in patient monitoring equipment, chip cards, electric razors, hearing aids, automobiles and industrial controllers. They are as small as a thumbnail or as large as a PC motherboard. They are developed in one-man businesses or in large industrial companies. EAGLE is being used in many of the top companies. The crucial reason for selecting EAGLE is not usually the very favorable price, but rather the ease of use. On top of that comes the outstanding level of support, which at CadSoft is always free of charge, and is available without restriction to every customer. These are the real cost killers!

EAGLE 4.1

Schematic Capture • Board Layout
Autorouter

Now available for Mac!

for Windows® Linux® Mac®

Version 4.1 Highlights

- Powerful library management: e.g. move devices between libraries, base library for packages, generate package variants from other libraries.
- Dynamic ratsnest during routing process.
- Copy function in schematic.
- Rotate components in 0.1-degree steps.
- Blind & buried vias and pads with off-center drill.
- User-defined background color.
- Miter function for (rounded) tracks.
- Smash for groups.
- Measure distances between arbitrary points.
- Choose alternative raster on-the-fly with Alt-key.

FREE

EAGLE 4.1 Light is Freeware!

You can use EAGLE Light for testing and for non-commercial applications without charge. The Freeware Version is restricted to boards up to half Eurocard format, with a maximum of two signal layers and one schematic sheet. All other features correspond to those of the Professional Version. Download it from our Internet Site or order our free CD.

If you decide in favor of the Commercial Light Version, you also get the reference manual and a license for commercial applications. The Standard Version is suitable for boards in Eurocard format with up to 4 signal layers (max. 99 schematic sheets). The Professional Version has no such limitations.

Prices	Light	Standard	Professional
Layout		199$	399$
Layout + Schematic		398$	798$
Layout + Autorouter		398$	798$
Layout + Schematic + Autorouter	49$	597$	1197$

Pay the difference for Upgrades

http://www.CadSoftUSA.com
800-858-8355

CadSoft Computer, Inc., 801 S. Federal Highway, Delray Beach, FL 33483
Hotline (561) 274-8355, Fax (561) 274-8218, E-Mail : info@cadsoftusa.com

block isn't erased every time a new graph is stored. The new data is written in the free space, and the flash memory block is completely erased only when it's filled by all 32 graphs. For visualization, a procedure seeks the last stored data and draws a graph.

CONSTRUCTION & RESULTS

I put all of the components on the front panel and connected it to the MCB2130 evaluation board with a flat ribbon cable. I modified the evaluation board by soldering an additional connector on the P0 empty position, and I used two metal corners instead of the plastic holders in order to have a mechanical junction with the front panel (see Photo 1a).

I can observe audio signals (or other signals) with a fine resolution (see Photo 1b). Of course, some additional functions (e.g., different trigger modes, a hardware input voltage divider (volt/div control), signal processing, and communication with PC) would be useful. I'll definitely consider adding these functions when I'm working on new applications (spectrum analyzers, EKG scopes, polygraphs, video games, and so on). As an example, you can press the F button to call the menu with reserved functions where the simplest screensaver is only implemented.

GENERATE VIDEO

With special test code, I measured the CPU load of the LPC2138 microcontroller with the video-generating task. It was approximately 21%. This means that the microcontroller can spend 79% of the time with another application.

I challenge you to build your own system. Good luck!

After graduating from the Moscow Aviation Institute (Russian Federation) in 1989, Ilya Mamontov worked as spacecraft electronic engineer for six years. He presently works in the atomic industry. His technical interests include designing, modernizing, repairing, and servicing measurement instruments. You may contact Ilya at illinoys@narod.ru. (Type "Circuit Cellar" in the subject line.)

PROJECT FILES

To download the code and additional files, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2006/196.

RESOURCES

T. Martin, "The Insider's Guide to the Philips ARM7-Based Microcontrollers," www.hitex.co.uk/arm/lpc2000book/.

NXP (founded by Philips), "LPC2131/2132/2138 User Manual," 2004.

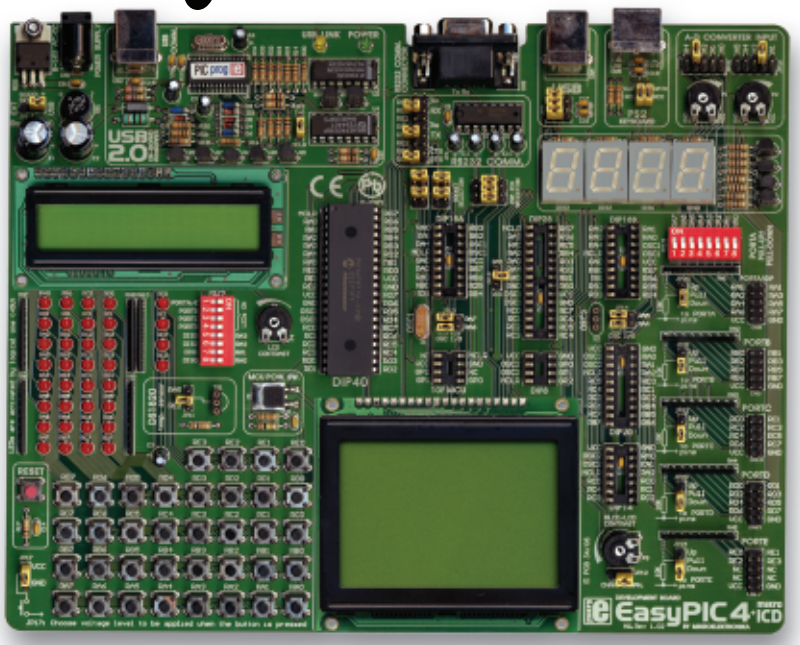
SOURCES

ARM7TDMI-S Microprocessor
ARM
www.arm.com

MCB2130 Evaluation board and µVision3 IDE
Keil
www.keil.com

LPC2138 Microcontroller
NXP (founded by Philips)
www.nxp.com

The Ultimate PIC Board
EasyPIC4 with mikro ICD



Feature-Rich PIC MCU Development System
-Hardware In-Circuit-Debugger
-Fast USB 2.0 In-Circuit Programmer
-PIC16F877A, LCD, GLCD, DS1820
-Free Versions of BASIC, Pascal & C Compilers
-CD with PICFlash2 Software & Code Examples
$169.00
SECURE ONLINE ORDERING....MAJOR CREDIT CARDS ACCEPTED
circuit-ED.com

Smart Probing

An R5F2113-Based Soil-Monitoring System

Alain's R5F2113-based monitoring system measures soil moisture, temperature, and light levels for potted plants. An audible alarm sounds when any measurement exceeds or falls below the preset limits. The principles he describes can be used in a variety of designs.

Have you ever purchased a healthy-looking plant from a local store and then found it dead only a few weeks later? Plants are propped up, doped with fertilizers, and grown in ideal conditions before they hit the stores. They then end up in the hands of people with busy lives who have a hard time remembering to take care of them on a regular basis. I've heard many people tell the tale of how their plants did not last long after they brought them home. I tried one of those $2 moisture sensors sold in stores, but the readings were all over the map. The device measured conductivity as an indication of soil moisture. I found it to be highly unreliable.

I've come up with a way to deal with this problem using electronics and modern technology. The challenge for a device of this kind was making sure it cost a lot less than the plant itself. At first, I considered developing an inexpensive disposable device, but I soon realized that I couldn't make such a device for less than $2. It made more sense to design a more expensive probe that could last at least a year or two and still save my plants.

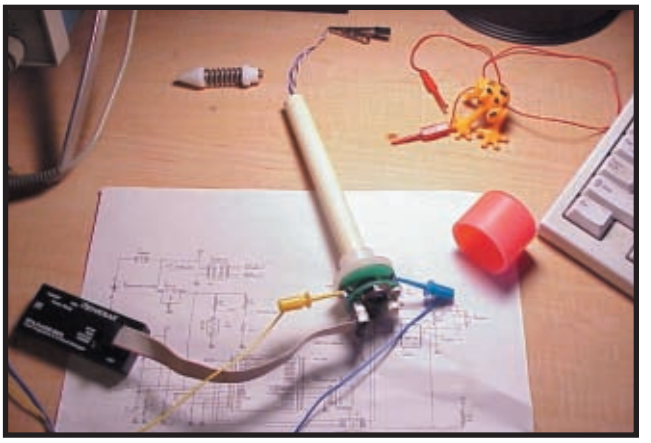


Photo 1—*The device is coded and debugged with the Renesas ICD. The pink plastic cover on the right protects the electronics. I used a crocodile clip to short the sensor wires to simulate maximum moisture on the moisture sensor (far left). Dry is open circuit. Do this to check the moisture circuit before casting plaster around the sensor's spring.*

In this article, I'll describe how I used a Renesas Technology R5F2113 microcontroller, a few inexpensive parts, simple measurement techniques, and some embedded software to build an effective soil probe (see Photo 1). The system works as a moisture and temperature alarm that helps you remember to take care of your plants.



Photo 2—*The probe is small enough to fit in a pot. You can hide the probe if you want it to be unnoticeable to passersby.*

PLANTS

Plants need air, light, water, and a horde of other things, including a comfortable temperature range, in order to grow properly. Unless you are growing plants in a vacuum or on the MIR Space Station, there isn't much that you can do about making sure a plant gets enough carbon dioxide.

$$CO_2 + 12H_2O + \text{light} \rightarrow C_6H_{12}O_6 + 6O_2 + 6H_2O$$

Too much or too little water can kill a plant. Another common cause of plant deterioration is a lack of sunlight, especially for indoor plants. Temperature is a no-brainer. Some plants can't tolerate the low temperatures, whereas some plants are hardy enough to survive frost. (Cacti tolerate high temperatures for long periods of time.) Using an electrical system to monitor these three key parameters should, in theory, enable you to reduce the chances of losing plants. But things are not always as simple as they seem.

Biologists classify and categorize living species of organisms. Plants are no exception. Scientific classification groups them in what is often called scientific taxonomy. Modern classification has its root in the work of Carolus Linnaeus, who grouped species according to shared physical character-

istics. There are numerous plant species out there, and each survives or deteriorates according to different combinations of the vital parameters I've described.

Most plants are subject to extremes in one form or another. Some plants can survive for weeks without water. A cactus can spend most of its day at 40°C and then encounter nearly freezing temperatures at night. My goal was to design a device that could be used to monitor the soil surrounding a plant of virtually any species.

CYCLES WITHIN CYCLES

My probe can monitor a plant's vital signs, and the data can be translated into a simple look-up table that can be adjusted according to species. On one side of this table are plants that are tolerant of extremes. On the other end of the table are plants that are very susceptible to minor extremes for short periods of time.

You can use a simple algorithm to monitor the plant's vital signs and use the table to determine when the measurements aren't so ideal. The device simply computes the amount of time that each parameter stays within predefined high and low limits. It then triggers an alarm when the thresholds are exceeded. That's essentially an integration of the error (the measurement minus some set point). It is then a question of finding the right set of values that work well for the species. This can be done empirically over extended periods of time by collecting and analyzing the measurements. The device enables you to tune the control table.

SMART PROBE OVERVIEW

My Smart Probe measures moisture, light, and temperature. It sounds an alarm when a measurement exceeds a certain threshold. The system is easy to use. Simply open the probe's cover, clip in a 3-V coin battery, and adjust the rotary switch settings according to the "Settings by Plant Species" guide. After closing the cover, insert the spiked end of the probe into the soil so that it is close to the plant's root system (see Photo 2). At this point, you can forget about the plant and go about your daily business. The probe will alert you when the plant needs attention.

The R5F2113 sits at the heart of the system, but you can use any microcontroller that supports the basic functions required by the design (see Figure 1). The implementation is split into two PCBs: a CPU board and an I/O board with connectors between them (see Photo 3, p. 64). A 3- to 3.6-V coin battery powers the R5F2113 (see Figure 2). An LDR, NTC, and a voltage reference are used to measure light and temperature. A vibration sensor and buzzer form the user interface. An ICD header provides a programming/debug interface. A three-pin header captures the measurements on a serial port.

To keep the overall cost of the system under control, I chose a very basic user interface to my probe. A piezoelectric buzzer generates musical tones as alarms. A vibration sensor and a BCD switch serve as input devices. I did not use a push button because waterproof buttons cost too much. A vibration sensor is more affordable and sealed in a rustproof can. The vibration sensor is essentially a switch that interfaces directly to the R5F2113's INT0 interrupt with a pull-up resistor.

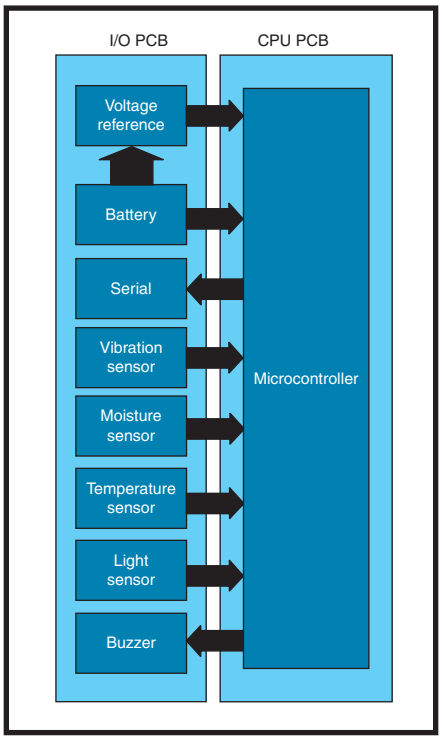


Figure 1—*The system is fairly simple. The voltage reference is for the ADC used to measure light and temperature.*

The piezoelectric buzzer (BZ1) is connected through a current-limiting resistor to the R5F2113's TZOut output to generate tones. These draw very little current, and P31 has enough drive capability to generate a loud buzz that you can hear from as far as 12′ away.

Buzzer tones generally annoy me. Fortunately, the R5F2113 has very flexible timers. I use these to generate musical tone sequences to represent different alarm conditions. I went through a lot of trouble to make it easy to create recognizable tunes. If you know a little bit about music, it should be painless to make any tunes you want spanning two octaves.

The device first starts up when you apply power. It immediately performs a built-in self test (BIST) and plays out the C major scale (Do Re Mi Fa So La Ti Do). If something wrong is detected at BIST time, the system plays the same scale in reverse, starting from a high pitched Do down to the Do one octave below.

BCD SWITCH

A BCD switch is used to select one of 10 different settings in a look-up table. Switch changes are polled for every second. The BCD switch is used to choose between the settings in Table 1 (p. 60).

Each column has a specific structure in the software implementation:

```
typedef struct LimitConfig
{
  Limit Light;
  Limit Moisture;
  Limit Temperature;
} LimitConfig;
```

`Limit` is defined as:

```
typedef struct Limit
{
  LEVEL LowLimit;
  LEVEL HighLimit;
  HOURS TimeLow;
  HOURS TimeHigh;
} Limit;
```

Less tolerant plants use a low

PIC-SERVO
MOTION CONTROL
MOTION CONTROLLERS FOR BRUSH, BRUSHLESS AND STEPPER MOTORS.
• controller chips
• controller boards
www.picservo.com
JEFFREY KERR, LLC

bob
basic overlay board
Introducing:
Vector graphics
Custom Font Capability
SPI or RS-232 Interface
Meet bob-4
Video display generator
www.decadenet.com
DECADE ENGINEERING
503-743-3194 Turner, OR, USA

Advertising Calendar
January
Issue #198
Embedded Applications
Space Close: Nov. 13
BONUS DISTRIBUTIONS:
DesignCon West
February, Issue #199
Wireless Communications
Space Close: Dec. 11
BONUS DISTRIBUTIONS:
APEC, IPC Printed Circuits Expo, APEX, and the Designers Summit
Call: Shannon Barraclough
(860) 872-3064
Shannon@circuitcellar.com

	BCD Switch position									
Measurement	0	1	2	3	4	5	6	7	8	9
Light										
Moisture										
Temperature										

Table 1— *The BCD switch selects between 10 ranges for light, moisture, and temperature measurements. Use lower switch settings for less tolerant plants and higher settings for more tolerant ones.*

switch setting, whereas more tolerant ones have a higher switch setting. The entries in the table have four parameters. `LowLimit` and `HighLimit` are the maximum and minimum thresholds for the measurements. `TimeLow` and `TimeHigh` are the times that measurements are allowed to be above and below the threshold limit.

CALIBRATION

The `LEVEL` and `HOURS` values, which must be determined empirically, are part of the device's calibration process. To calibrate, place the BCD switch in position 9 and the device will send the raw measurements to a serial port on the R5F2113. Use simple serial terminal software at 9,600 bps, no parity, 8 data bits, and 1 stop bit to capture the readings and import them to a spreadsheet to graph and analyze the data. The format is a comma-delimited string of three numbers (temperature, light, and moisture) followed by 0x0d, 0x0a. The temperature is in degrees Celsius. Other readings are the raw values from the 10-bit ADC. Based on the results, use your judgment to pick appropriate values for the table.

Take the light measurement, for example. A low light setting for BCD position 0 would allow the plant to be at low light levels or high light levels only for limited periods of time over the course of a day (say, 6 h below the 30% level and 8 h above 70%). To think about these settings is to think about how long you want to expose the plant to the set extremes in terms of hours.

PRINCIPLES OF OPERATION

Except for a few unique species of plants (e.g., some types of bamboo), plant growth is a slow process. My probe measures the soil's moisture, temperature, and light level every hour. The cycle time is configurable. When the moisture, light, or temperature reaches a preset level and remains there for longer than a set amount of time, the system sounds an alarm.

The main loop simply configures the device, performs BIST, and then goes into a loop to take measurements, update the alarm's status based on those measurements, and send the measurements over the serial port. Once you apply power, the probe takes measurements, typically every hour, and totalizes time to 1-s resolution. When it's finished, it remains silent, but you can tap it gently and activate the vibration sensor to play out the status. In response, the system beeps and sends status information in the form of tones: light, moisture, and temperature, in that order. Two tones are used to represent a measurement level. One tone represents a reading above the high limit (<HIGH>). The other is for a reading below the low limit (<LOW>). If everything is well when you tap the device, you will hear the following:

<BEEP><nothing><nothing><nothing>

If the light, moisture, or light readings are at an odd level, you'll hear the following sort of response:

<BEEP><LOW><HIGH><LOW>

The tones I used were pretty lame because I didn't want to deal with the copyright issues associated with using popular tunes. However, that does not stop you from making your own tunes.

MOISTURE MEASUREMENT

The system doesn't need to take exact measurements because you are dealing mostly with extremes. The idea is to take light level, moisture, temperature, and time measurements as cheaply as possible. Measuring light levels on the cheap involves a light-dependent resistor (LDR). There are numerous options for measuring the temperature, but because the R5F2113 has an ADC, a thermistor is good

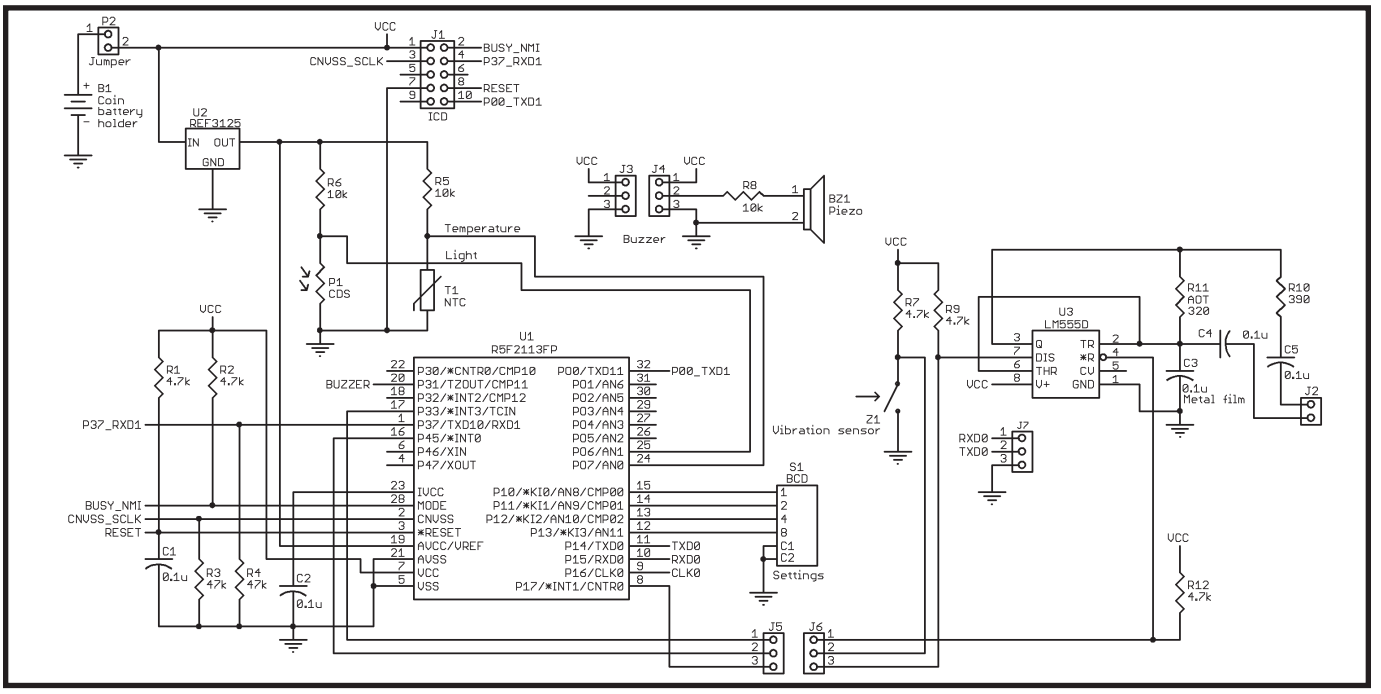
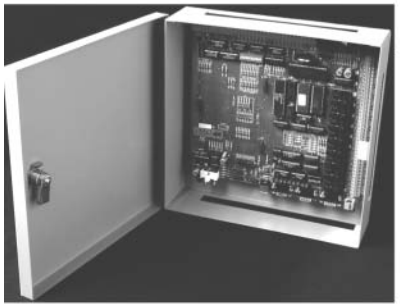


Figure 2—*At the heart of the probe is a Renesas Technology R5F2113 microcontroller. The circuit is implemented on two separate PCBs.*

enough. Measuring moisture, however, is not as easy. My low-cost design uses the gypsum block to measure soil moisture. It's a lot better than conductivity meters.[1]

The gypsum block (plaster, CaSO4) measures soil tension, which is related to soil moisture. The AC resistance of the block changes with moisture tension. By adding a gypsum block to an RC timing network, the resistance of the gypsum block is measured

HomeVision-Pro



CONTROLS:
- Lighting
- HVAC
- Security
- A/V
- Much more...

HomeVision-Pro is the newest, most powerful automation controller in the award-winning HomeVision® product line. It's configured from a PC, but can then run standalone. Features:

- Two-way X-10 & IR
- Video screens on a TV
- 8 relays
- 16 digital & 4 analog I/O
- 64 digital temp sensors
- 1 USB & 3 serial ports
- Powerful scheduling (>8000 lines, >2000 objects)
- Clock
- PC web server
- Expandable
- Software included
- 3-year warranty

Custom Solutions, Inc.
800-398-8882, Web: www.csi3.com, E-mail: csi@csi3.com

EMBEDDED LINUX
Made Easy!



Turn-Key Solutions!
- ECLIPSE IDE with full Library and Module Support
- Integrated GDB Source Level Debugger
- Resident Menu Driven Setup Utility
- No Charge for Tools or Standard Linux Install
- Free Integration, Setup and Test

2.6 Kernel

Since 1985 OVER 21 YEARS OF SINGLE BOARD SOLUTIONS

EMAC, inc.
Phone 618 - 529 - 4525 Fax 618 - 457 - 0110
2390 EMAC Way, Carbondale, Illinois 62902
World Wide Web: http://www.emacinc.com

indirectly by measuring the frequency.

The measuring circuit is built around a Texas Instruments TLC555 timer, which is a low-power, low-operating-voltage variant of the common LM555. The frequency of this circuit is measured through pin INT3/TCIN to 16-bit resolution using TimerC in Capture mode and the internal high-speed oscillator as a reference. The frequency of the gypsum block is around 53 Hz when dry and 230 kHz when wet. That's quite a wide dynamic range.

TEMP MEASUREMENT

Temperatures are measured with a thermistor connected to a voltage reference. The thermistor connects to an ADC input on the microcontroller. The temperature (Kelvin) is a function of resistance:

$$T_K = \frac{1}{A + B \times \ln\left(\frac{RT1}{R5}\right) + C \times \ln\left(\frac{RT1}{25}\right)^2 + D \times \ln\left(\frac{RT1}{R5}\right)^3}$$

where *R5* (10 K) is the resistance at 25°C.

For the thermistor (the same one that comes with the R8C demonstration board), the coefficients *A*, *B*, *C*, and *D* have specific values: $A = 3.354016e^{-3}$, $B = 2.569107e^{-4}$, $C = 2.626311e^{-6}$, and $D = 0.675278e^{-7}$.

The temperature (Celsius) is:

$$T_C = T_K - 273.15$$

The software has routines to calculate the temperature using T_C or a look-up table. The spreadsheet used to calculate the look-up values is posted on the *Circuit Cellar* FTP site.

TIME MEASUREMENT

The probe must determine moisture, light, and temperature levels. In addition, it must determine how long these levels are maintained. Thus, the system includes a timing feature.

The R5F2113's timers are fairly accurate. They can measure time to 0.5 s or less resolution. The software maintains variables to keep track of hours, minutes, and seconds. It implements a simple software real-time clock (RTC). Only relative time is required to integrate measurements, so the system doesn't keep track of dates. TimerY drives the software RTC.

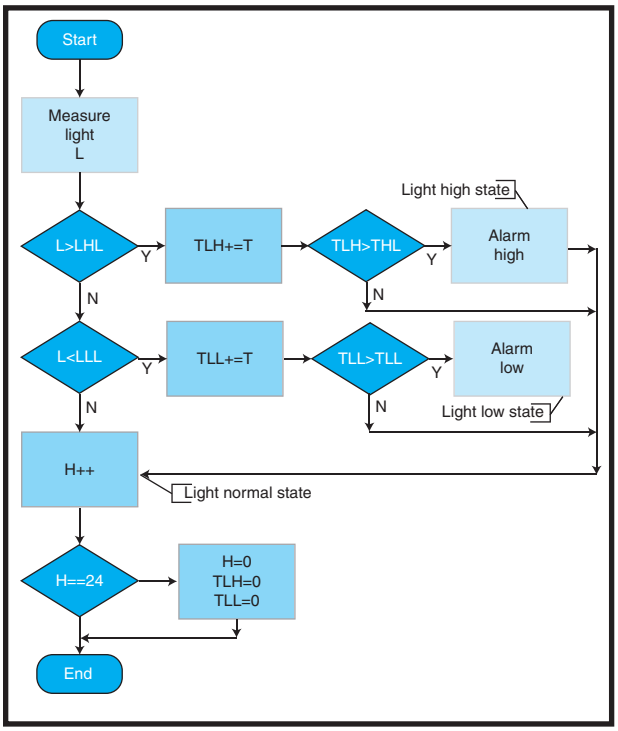


Figure 3—*Check out the flowchart for the alarm logic. It calculates the amount of time that measurements stay above or below set thresholds over a 24-h period.*

MUSIC

The R5F2113's Y and Z timers both have programmable waveform generation capabilities. To keep the component count and costs low, an external crystal isn't used as a clock reference. The internal high-speed oscillator provides all of the clocks. TimerZ provides the basic capabilities required to generate musical tones. TimerX provides the timing for beats. With notes and beats, it's easy to make music.

Three parameters control the frequency of the waveform generated on the TZOut output pin: the prescalar register value (n), the primary register (m), and the secondary register (p). These are used to generate tones of known frequencies starting from concert A (440 Hz) and one octave above. An Excel spreadsheet calculates the appropriate n, m, and p values.

The software sets these values to play notes in an interrupt-driven manner with minimal intervention from the CPU. The CPU can enter a wait state and the timers will still generate music. The CPU wakes up from a wait state to select the next note. The software is written to make it easy for the device to play recognizable tunes. Even if you have only a rudimentary knowledge of music, it is easy to compose tunes. You can assign different tunes for different alarm conditions.

ALARM ALGORITHM

All measurements use a simple alarm algorithm (see Figure 3). As an example, let's consider light level measurement. The light level (L) is measured every hour (H). TLH, TLL represent the hours for which the light level are low and high, respectively. High and low light levels are defined as LHL and LLL, respectively. THL and TLL are the time limits for the high and low measurements. Measurements are reset every 24 h.

SYSTEM DEVELOPMENT

With such a simple interface, every plant owner will find it easy to use my probe. This design is based on the R5F2113, but I'm sure you could use a different microcontroller. I detailed everything you need to know about taking measurements, playing music, and running the RTC in high-level C language. Now it's your turn to get started on your own project

You can buy a gypsum block for about $8, or you can try to make your own like I did. A gypsum block is nothing but inexpensive plaster that you can buy at your local hardware store. Commercial blocks are made in batches of highly controlled ingredients to achieve consistency and repeatability when calibrating. For the purposes of this application, however, a homemade block works well. Be sure to use stainless steel electrodes to avoid rusting. I used a stainless steel

Support Across The Board.™



Bringing Products to Life with Mixed-Signal FPGAs.

At Avnet Memec, support across the board is much more than a tagline for us. From initial design through end of life — we are deeply committed to driving maximum efficiency throughout the product lifecycle.

The Challenge

How do you launch the first **Mixed-Signal FPGA** to the engineering masses? Actel's Fusion Programmable System Chip integrates configurable analog, large Flash memory blocks, comprehensive clock circuitry, and high performance programmable logic in a single chip. With so much to offer, the company wanted to quickly reach the engineering market. Enter Avnet Memec.

The Solution

Actel's leading distribution partner Avnet Memec underwent rigorous product training on the applications and benefits of Fusion. In the spring of 2006, Avnet Memec rolled out its SpeedWay™ full-day hands-on workshops and labs, highlighting the versatility of Fusion for OEM engineers. All this adds up to quite an education.

As OEMs look to implement Fusion into their designs, Avnet Memec and Actel will be there all the way — providing technical support in every phase of design development. It's true support across the board.

To learn more, request a free Fusion informational CD and enter to win one of **10 free Fusion Starter Kits**, go to:
www.em.avnet.com/actel/satb

Actel® FUSION ONECHIP

AVNET® Memec

Enabling success from the center of technology™

1 800 408 8353
www.em.avnet.com

Avnet Green Initiative

Supplier Authorized Distributor

© Avnet, Inc. 2006. All rights reserved. AVNET is a registered trademark of Avnet, Inc.©Actel 2006. All rights reserved. Actel logo and combinations thereof, and others are the trademarks of Actel Corporation or its subsidiaries.

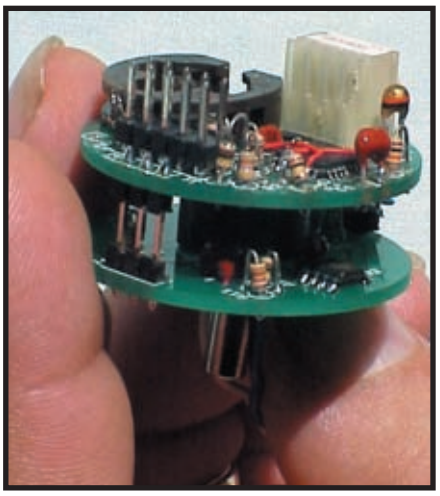


Photo 3—*The probe is implemented as two separate PCBs. The bottom PCB is for moisture measurements. The top PCB is for the microcontroller, ICD header coin battery, and BDC switch. The light and temperature sensors are on the top board. The device's cover is made of translucent plastic so that light can reach the light sensor.*

spring and screw as electrodes.

I made my probe on a small lathe, hence its size and shape. You can build a smaller, more attractive probe if you want.

You should find it fairly easy to add to or alter the C language code for this project. I recently updated the code. I also added serial communications to facilitate calibration. Visit www.techboxes.com/smartprobe for information about tools written in .NET 2.0 that can capture and log measurements for analysis.

Alain d'Espaignet holds a national diploma in electronic engineering instrumentation and a Bachelor's degree in computer science. He is a software engineer who makes games at Electronic Arts. You may contact him at adespaignet@ea.com and adespaignet @techboxes.com.

PROJECT FILES

To download the code and additional files, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2006/196.

REFERENCE

[1] "Gypsum Blocks/Electrical Resistance," www.sowacs.com/sensors/gypsum.html.

RESOURCES

Measurement Engineering Australia, "Resurrecting the Gypsum Block for Soil Moisture Measurement," www.sowacs.com/feature/mea/mea.html.

———, "The Gypsum Block," www.mea.com.au/products/Soil/GB/gb.html.

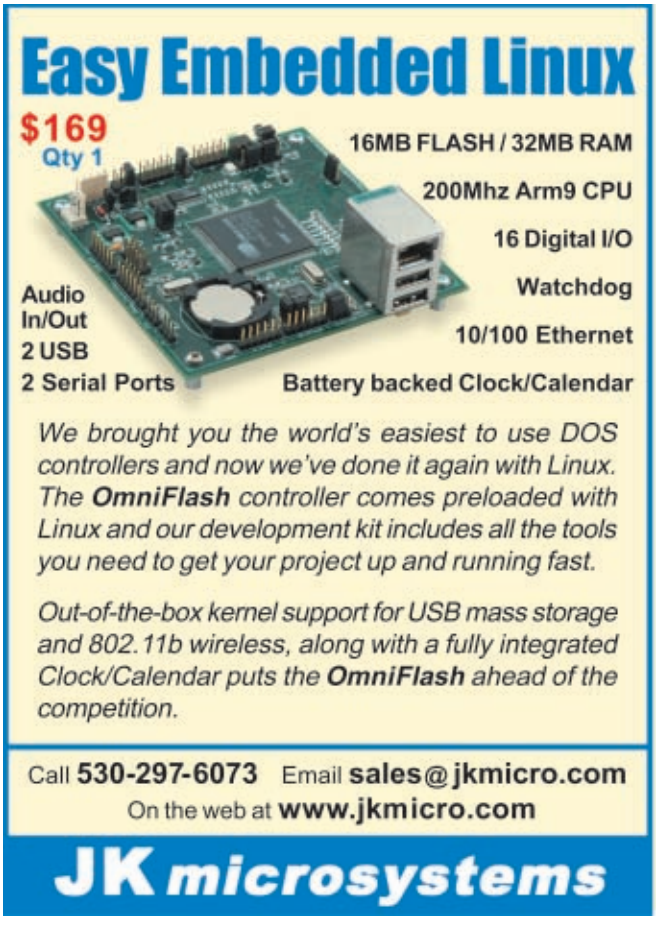
Renesas Technology Corp., "R8C Tiny," http://america.renesas.com/fmwk.jsp?cnt=r8ctiny_series_landing.jsp&fp=/products/mpumcu/m16c_family/r8c_tiny_series/.

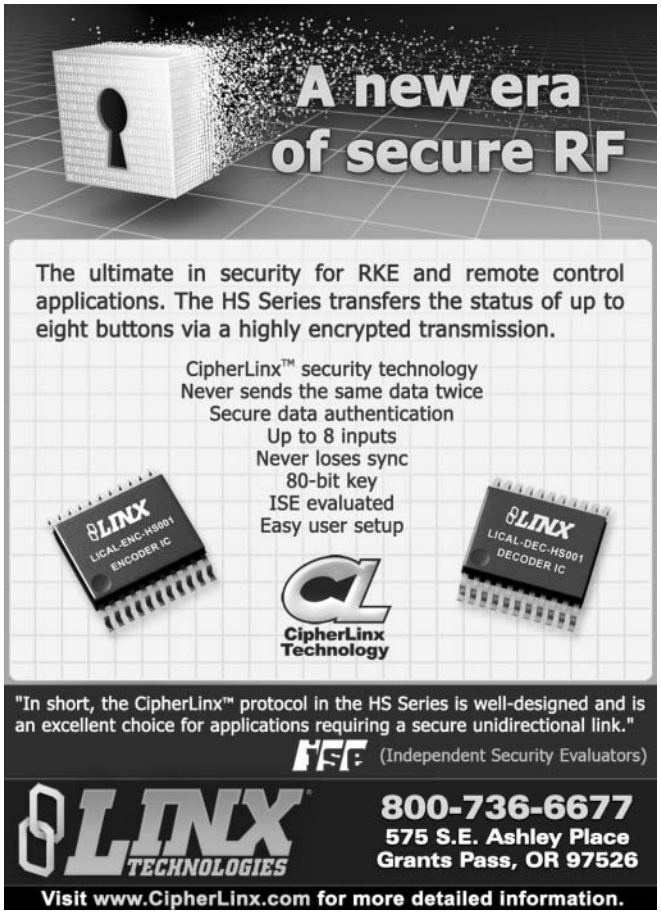
SOURCES

R5F2113 Microcontroller
Renesas Technology Corp.
www.renesas.com

TLC555 timer
Texas Instruments, Inc.
www.ti.com

NTC Thermistors
Vishay Intertechnology, Inc.
www.vishay.com

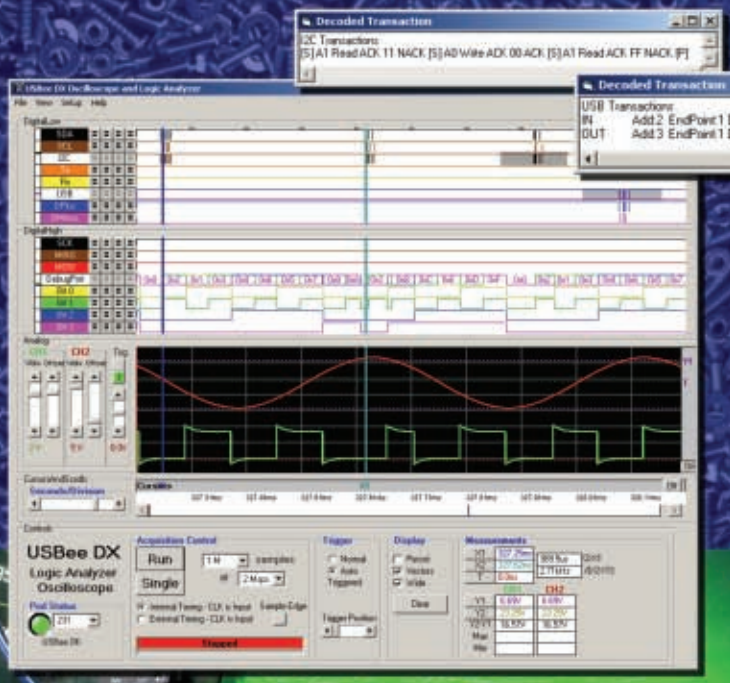




USBee DX

Oscilloscope/Logic Analyzer

2 Analog and 16 Digital Signals
Up to 24Msps, 100+ Million Samples
Dual 8-bit ADC, +/-10V inputs



Bus Decoding

Click-and-Drag Instant Decode of Bus Transactions
I2C, SPI, ASYNC, CAN, USB Low and Full Speed
I2S, SM Bus, 1-Wire, PS/2

Digital Voltmeter

2-Channel, +/-10V, 8-bit ADC, Logging

Data Logger

2 Analog and 16 Digital Signals Plus Timestamp

Digital Signal Generator

16 Digital Outputs, Up to 24Msps
Playback of Logic Analyzer Traces

Pulse Width Modulator

16 User Controlled PWM Channels

Frequency Counter

16 Channel Counter to 24MHz

Frequency Generator

Generates Sets of Common Frequencies

I2C Controller

Control I2C Devices Using Simple Text Scripts

Remote Controller

Easily Control Your Hardware Using Your PC

Pulse Counter

16 Channel Pulse Counter With Gating Control

plus the USBee Toolbuilder Source code and Library

Create Your Own Applications to Control the USBee DX



Actual Size

Data Extractors

Optional Software Modules for the USBee AX-Pro and USBee DX

Continuous Real-Time Embedded Bus Data Streaming

Store Bus Data to Disk or Send to Your Custom Application

Capture and Process Entire Test Sequences

Parallel, Serial, I2C, USB, ASYNC, CAN, SMBus, SPI, I2S, 1-Wire



USBee.com (951) 693-3065
support@usbee.com



Stand-Alone Ethernet Control

Ethernet technology is constantly evolving. Fortunately, you have Fred to explain what's new before you leap into your next single-IC Ethernet project.

I recently had the pleasure of keeping some of you captive in a classroom at the Microchip Technology MASTERs 2006 conference in Scottsdale, Arizona. This, year I presented classes on both 802.11b and the new Microchip ENC28J60 Ethernet engine. We had some fun with EASY buttons, networking hardware, and moonshine.

If you elected to sit in one of my ENC28J60 classes, you were subjected to elements of my little "garage stack" that are aimed at the ENC28J60 silicon. Talking about the code and hardware you've created for the business end of three days tends to spring forth ideas for improving the efficiency of the system you're constantly chattering about. In my quest to simply get my little garage stack off the ground, I totally ignored one of the most important features of the ENC28J60: the direct memory access (DMA) subsystem. In this article, I'll talk about the ENC28J60 DMA subsystem and apply it to my existing mini-stack. I'll also look at how to zip up the performance of any ENC28J60 implementation.

My excursions to the MASTERs have always been exciting and informative. This year, I was introduced to the new Microchip PIC18F97J60, which is the next logical step in the evolution of the ENC28J60 technology. I'll be sure to commit some space in this column to the PIC18F97J60 and the associated development kit. With all of that, let's get started and take a closer look at the ENC28J60 DMA subsystem.

ENC28J60 DMA ENGINE

Before I begin this DMA discussion, I've got to let out a "yeehaw"! The 8- to 10-MHz SPI clocking restriction associated with the initial production releases of the ENC28J60 are nevermore in the new B5 silicon. Even with that news, the words you'll absorb concerning the ENC28J60 DMA are still relevant no matter which revision of ENC28J60 silicon you're designing into your project.

The ENC28J60 is equipped with a dual-purpose DMA engine. In one instance, the ENC28J60's DMA subsystem can be used to copy data between points inside the ENC28J60's internal 8 KB of buffer SRAM. The other pleasant surprise that comes standard with the ENC28J60's DMA engine is its ability to calculate standard 16-bit checksums used in TCP segments and IP datagrams using the DMA hardware.

I was very skeptical of the DMA's operation when I first began to use the ENC28J60 in a serious manner. The problem wasn't at all related to the ENC28J60 hardware. The grind was with me. I really had not taken time to completely understand the ENC28J60 DMA subsystem. So, instead of taking the time to get with the DMA program, I chose to simply ignore it. That was a bad decision on my part. As I grew more familiar with the quirks associated with the ENC28J60, my garage stack code performance improved accordingly.

My first run of stack code for the ENC28J60 produced a PING response time that averaged between 40 and 50 ms. Even though everything worked as designed, the PING performance was very poor. So, I chopped and hacked at the garage stack source code and managed to get the PING times down to an average of 15 ms.

I was able to achieve the 15-ms PING response times by using SPI bursting. In my original garage stack code, I repeatedly called the read register and write register SPI routines for every byte of data that was passed between the host microcontroller and the ENC28J60. As you can imagine, the computational cost of calling the read and write register routines was very high.

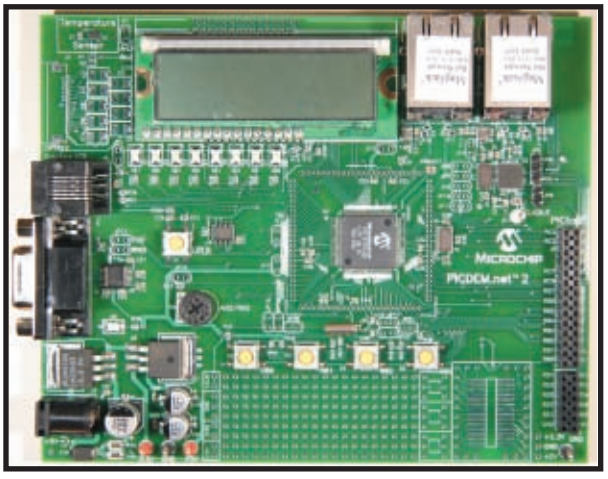


Photo 1—*Just in case having a couple of Ethernet ports isn't enough for you, the pad area at the lower right supports an ENC28J60 in the 28-pin SOIC package.*

The ENC28J60 silicon was in tears asking me to heed to obvious. The obvious in this case was the ENC28J60's ability to automatically increment its internal SRAM read/write pointers. After the initial read SRAM or write SRAM command is issued via the SPI interface, the ENC28J60 SRAM access mechanism will automatically increment its internal SRAM pointer until the SPI operation is terminated with the lifting of the active low chip select (*CS) signal associated with the SPI interface. By simply bursting the SPI reads and writes, I trimmed 30 ms off the PING response times, and I haven't even implemented any DMA functionality yet.

At this point, no matter what I did to the source code, I couldn't slice off any significant amount of time from the PING response of 15 ms. So, I scanned the ENC28J60 garage stack to see which routines I was calling most often. As it turns out, every outgoing frame gets at least one checksum calculation. My garage stack's PING reply function is based on ICMP, which requires yet another checksum calculation. A working example of my checksum code is shown in Listing 1.

If you've read through any of my books, you know that I like to use arrays instead of pointers to illustrate how things work inside the Internet protocols. That way I can get through to the beginner and still convey the overall idea to a more advanced coder. Using the array approach is not the most efficient method of working through a stack, but that's what comes with the package. With that, note that I clear the IP header checksum fields in my packet array before setting up to call the checksum routine. The checksum code in Listing 1 produces a checksum that is the 16-bit one's complement of the one's complement sum of all 16-bit words in the header. Here's where I decided to attack the PING response time bogey with a DMA smart bomb.

The ENC28J60's DMA engine's checksum calculation logic sees the bytes of data within the ENC28J60's SRAM as a series of 16-bit big Endian integers. The resulting checksum that is generated by the DMA checksum calculation hardware is identical to the checksum generated by my code in Listing 1, the 16-bit one's complement of the one's complement sum of all 16-bit words in the selected data field.

I like to think of the ENC28J60 as a PIC because the ENC28J60, like the PIC, is controlled via a banked set of registers. The beginning byte of a data area targeted by the ENC28J60's DMA engine is defined by the values that are loaded into the high and low bytes of the ENC28J60's EDMAST (DMA Start) register pair. The ending byte of the DMA's operational data area is also delineated by a DMA End register pair (EDMAND). After the checksum data boundaries are defined via the register pairs, all that is needed to kick off the hardware checksum calculation process is the setting of a couple of bits in the ENC28J60's ECON1 register, CSUMEN (Checksum Enable), and DMAST (DMA Start). When the checksum has been generated, the ENC28J60's checksum hardware will clear the DMAST bit. If your application requires interrupt capability, you may enable the ENC28J60's DMA interrupt mechanism, which will set the DMA interrupt flag (DMAIF) and generate an interrupt if so desired upon the completion of the checksum calculation process. The resulting hardware-generated 16-bit checksum value is placed into the ENC28J60's DMA checksum register pair, EDMACSH and EDMACSL, for use by the application. There are no side effects to using the DMA engine in checksum mode as no DMA pointers are altered and nothing is written to the memory area used in the checksum process. The ENC28J60's DMA checksum calculation engine also automatically pads odd-byte-length data areas with 0x00 to maintain the image of 16-bit integers within the designated data memory area.

The DMA checksum calculation engine runs at one clock cycle per byte used in the checksum calculation. With the ENC28J60 clock running at

Listing 1—*This is a typical spin of code involved with performing a checksum calculation. This particular code snippet is used to generate an IP header checksum.*

```
//This is a typical setup to call the cksum() routine
//Calculate the IP header checksum
packet[ip_hdr_cksum]=0x00;
packet[ip_hdr_cksum+1]=0x00;
hdr_chksum =0;
hdrlen = (packet[ip_vers_len] & 0x0F) * 4;
addr = &packet[ip_vers_len];
cksum();
chksum16= ~(hdr_chksum + ((hdr_chksum & 0xFFFF0000) >> 16));
packet[ip_hdr_cksum] = make8(chksum16,1);
packet[ip_hdr_cksum+1] = make8(chksum16,0);
*****************************************
CHECKSUM CALCULATION ROUTINE
*****************************************
void cksum()
{
    while(hdrlen > 1)
    {
        data_H=*addr++;
        data_L=*addr++;
        chksum16=make16(data_H,data_L);
        hdr_chksum = hdr_chksum + chksum16;
        hdrlen -=2;
    }
    if(hdrlen > 0)
    {
        data_H=*addr;
        data_L=0x00;
        chksum16=make16(data_H,data_L);
        hdr_chksum = hdr_chksum + chksum16;
    }
}
```

25 MHz, performing a checksum operation on an IP header containing 20 bytes would take approximately 0.8 µs. Calculating the checksum pair totaling 60 bytes inside a typical PING response with 32 bytes of data would take 2.4 µs.

The entire process of calculating the IP header checksum and placing the checksum bytes into the PING reply package is shown in Listing 2. The IP header length of 20 bytes is determined by parsing the IP version/length field, which contains 0x45 in this case. At this point, the meat of the PING response is already loaded into the ENC28J60's transmit buffer, which is allocated within the ENC28J60's 8 KB of internal SRAM buffer space. Data areas inside the current Internet protocol headers are always located at the same offsets within the headers. You need to calculate the IP header checksum beginning with the IP version/length field. So, you must set up the DMA starting point at the first IP version/length byte. The 1 added to the `offsetval` variable is actually the control byte required by the ENC28J60. The control byte is always the first byte within the transmit buffer space. TXSTART defines the beginning offset of the transmit buffer within the ENC28J60's 8-KB buffer area, and the variable `ip_vers_len` is defined as the offset of the IP version/length field within the transmit buffer. The combination of `1+TXSTART+ip_ver_len` produces the offset into the transmit buffer you will use to begin your DMA checksum calculation. The end of the DMA checksum calculation byte area is 20 bytes down the road, and this value is loaded into the DMA End register pair. The `CSUMEN` and `DMAST` bits are set and the routine spins until the `DMAST` bit clears. The resulting 16-bit checksum value is then bursted into the IP header checksum fields via the SPI interface.

When the PING responses were moving along at an accelerated pace, I should have been happy. But I got greedy. The next point of attack within my garage stack was the next most time-consuming task of writing values from the host microcontroller's packet array into the ENC28J60 transmit buffer area.

Listing 2—*At this juncture, the entire ICMP response has been loaded into the ENC28J60's transmit buffer area. When the IP header and ICMP header checksums are generated and loaded, the PING response is then transmitted.*

```
//Compute ip header checksum
hdrlen = (packet[ip_vers_len] & 0x0F) * 4;
offsetval = 1+TXSTART+ip_vers_len;
wr_reg(EDMASTL,LOW_BYTE(offsetval));
wr_reg(EDMASTH,HIGH_BYTE(offsetval));
offsetval += hdrlen;
wr_reg(EDMANDL,LOW_BYTE(offsetval));
wr_reg(EDMANDH,HIGH_BYTE(offsetval));
bfs_reg(ECON1,ECON1_CSUMEN | ECON1_DMAST);
while(rd_reg8(ECON1) & ECON1_DMAST);
chksum_lo = rd_reg8(EDMACSL);
chksum_hi = rd_reg8(EDMACSH);
//Write ip checksum values to TX buffer
offsetval = 1+TXSTART+ip_hdr_cksum;
wr_reg(EWRPTL,LOW_BYTE(offsetval));
wr_reg(EWRPTH,HIGH_BYTE(offsetval));
enc_spi_on;
SSP1BUF = WBM;
while(SSP1IF == 0);
SSP1IF = 0;
SSP1BUF = chksum_hi;
while(SSP1IF == 0);
SSP1IF = 0;
SSP1BUF = chksum_lo;
while(SSP1IF == 0);
SSP1IF = 0;
enc_spi_off;
```

The hardware and IP addresses must be flipped when building the PING response. I had that under control because I was bursting the address values I had to read from the incoming packet out of the SPI interface directly into the ENC28J60's transmit buffer. The 32 bytes of data within the incoming PING packet were not touched and simply echoed back. So, instead of reading those values into the packet array and then spitting them back out the SPI, I decided to put the DMA to work as a copier.

I read the ENC28J60's receive buffer contents up to the data area of the ICMP PING packet (see Listing 3).

Listing 3—*Using the ENC28J60's DMA engine as a copier is easy. All you do is define the beginning and ending bytes to be copied and designate a copy-to destination offset. Clear the `CSUMEN` bit, set the `DMAST` bit, and wait for the dust to settle.*

```
//Get the rest of the receive buffer contents
banksel(ERDPTL);
i = rd_reg8(ERDPTL);
j = rd_reg8(ERDPTH);
wr_reg(EDMASTL, i);
wr_reg(EDMASTH, j);
++rx_end;
wr_reg(EDMANDL,LOW_BYTE(rx_end));
wr_reg(EDMANDH,HIGH_BYTE(rx_end));
i = rd_reg8(EWRPTL);
j = rd_reg8(EWRPTH);
offsetval = make16(j,i);
--offsetval;
wr_reg(EDMADSTL,LOW_BYTE(offsetval));
wr_reg(EDMADSTH,HIGH_BYTE(offsetval));
bfc_reg(ECON1,ECON1_CSUMEN);
bfs_reg(ECON1,ECON1_DMAST);
while(rd_reg8(ECON1) & 0x20);
```

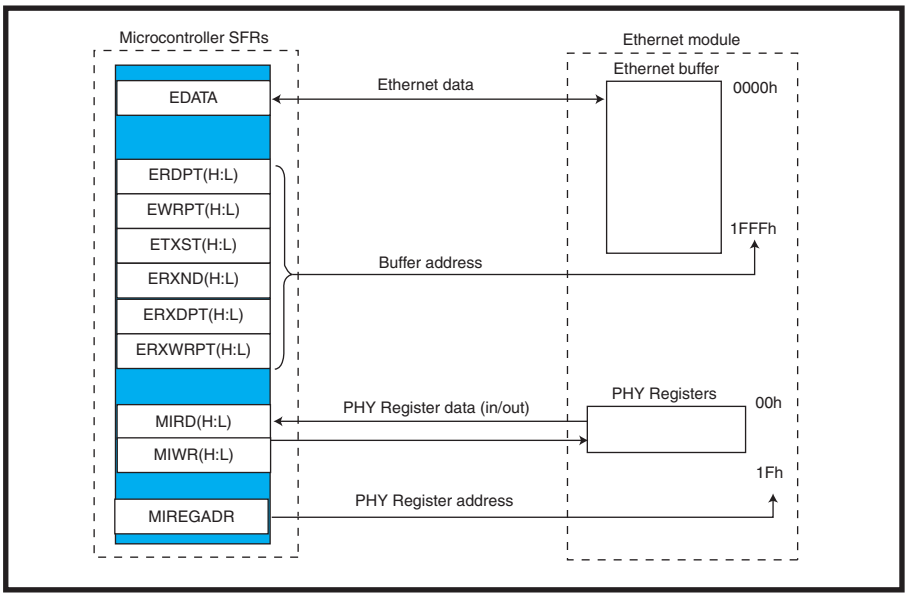


Figure 1— *The PIC18F97J60 Ethernet registers are identical in name and function to the ENC28J60's registers. The significant difference between the architectures is the elimination of the SPI data interface in the PIC18F97J60. The legacy SPI data conduit found within the ENC28J60 is replaced with the implementation of the EDATA register in the PIC18F97J60.*

That left my receive buffer read pointer register pair (ERDPTL/ERDPTH) pointing at the 32 bytes of data at the end of the incoming PING packet. I loaded the contents of the receive buffer pointer into the DMA Start register pair. This was where I began my DMA copy. Earlier, I had calculated the offset at the end of the incoming PING packet. The rx_end value I came up with had to be incremented by one to account for the transmit buffer control byte. This ending value was then loaded into the DMA End register pair (EDMANDL/EDMANDH). The copying stopped there. The extent of my SPI bursting into the ENC28J60's transmit buffer area was tracked by the write pointer register pair (EWRPTL/EWRPTH), which denoted the destination in which the copied bytes had to be placed. I noticed in my debug runs that an extra byte was skipped when I executed the copy. So, assuming the DMA copy increments before the write, I set the offset back a byte before kicking off the DMA memory copy by setting the DMAST bit. Note that the CSUMEN bit is cleared for a DMA copy operation. As with the DMA checksum calculation, the DMAST bit is cleared when the smoke settles. With the inclusion of simple ENC28J60 DMA mechanics in my little garage stack, PING response times are now measuring in at an average of 1 ms.

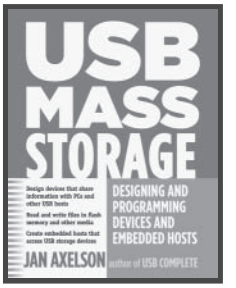
THE NEXT LEVEL

Although the ENC28J60 part works very well (the aggregate data rate is 8.9 Mbps with a 10-MHz SPI clock), the SPI interface is the bottleneck if you have to choose a weak point. There's only one way to completely eliminate the need for a SPI communications channel: put everything (host and Ethernet engine) on the same slice of silicon.

The PIC18F97J60 is a full-blown PIC microcontroller coupled with a full-blown ENC28J60 in a single package. If you've been following my garage stack discussions, the ENC28J60 Ethernet implementation learning curve will be zero because there are PIC18F97J60 special function registers (SFRs) identical to the ENC28J60 registers assigned to control the flow of data to and from the PIC18F97J60's integrated Ethernet engine. If you take a look at Figure 1, you'll see familiar register pair names such as ERDPT and EWRPT. You'll also note a new register called EDATA.

The EDATA register replaces the SPI interface on the PIC18F97J60. To feel the impact of what the EDATA register implementation does for you, peruse Listing 4 (p. 70). The ERDPT register pair (ERDPTH/ERDPTL) is

ADD MASS STORAGE
TO YOUR DESIGNS



USB Mass Storage
Designing and Programming Devices and Embedded Hosts
Jan Axelson
ISBN 1-931448-04-3 $29.95
Lakeview Research LLC www.Lvr.com
From the author of USB Complete

USB/ETHERNET DAQ
LabJack
UE9



Available now for only ...
$429 qty 1
USB/Ethernet Data Acquisition & Control

* USB 2.0/1.1 and Ethernet
* 14 analog inputs (12- to 16-bit)
* Stream input data up to 50 kHz
* Use with C, VB, LabVIEW, etc.
* Includes DAQFactory Express
* Operates from -40 to +85 deg C
* 2 analog outputs (12-bit)
* 23 digital I/O
* Up to 2 counters (32-bit)
* Up to 6 timers
* Approx. 3" x 7" x 1"

LabJack Corporation, Colorado, USA
info@labjack.com, (303) 942-0228
www.labjack.com

ANDRE LAMOTHE'S
XGAMESTATION
DO-IT-YOURSELF VIDEO GAME SYSTEM
Inspired by the Atari 2600, Apple II & Commodore 64!



INCLUDES:
• Assembled XGS Micro Edition Unit!
• Complete Development Kit!
• Tools, Demos & Utilities!
• eBook on Designing the XGS Console!
• Cables and Power Supply Included!





WWW.XGAMESTATION.COM
(925) 736-2098 SUPPORT@NURVE.NET

Listing 4—*What do you want me to say?*

```
char rd_reg8(char addr)
{
        enc_spi_on;
        SSP1BUF = RCR | addr;
        while(SSP1IF == 0);
        SSP1IF = 0;
        SSP1BUF = 0;
        while(SSP1IF == 0);
        SSP1IF = 0;
        enc_spi_off;
        return (SSP1BUF);
}
char rd_reg8_97J60(void)
{
        return EDATA;
}
*********************************
void wr_reg(char addr, char data)
{
        enc_spi_on;
        SSP1BUF = WCR | addr;
        while(SSP1IF == 0);
        SSP1IF = 0;
        SSP1BUF = data;
        while(SSP1IF == 0);
        SSP1IF = 0;
        enc_spi_off;
}
void wr_reg_97J60(char data)
{
    EDATA = data;
}
```

used to point to the desired byte to be read from the PIC18F97J60's Ethernet SRAM buffer, which just happens to look just like the ENC28J60's 8-KB SRAM buffer. By reading the EDATA register, the value pointed to the ERDPT register pair is passed to the application. Writing data bytes to the PIC18F97J60's 8 KB of SRAM buffer area is just as easy. The EWRPT pointer pair illuminates the target SRAM, and a simple write to the EDATA register puts the data on target.

It seems that every rev of Microchip microcontrollers gets faster and faster. The PIC18F97J60 must use a 25-MHz primary clock to service the enclosed ENC28J60. That means that when you run the 25-MHz clock input through the PIC18F97J60's PLL you can get a maximum clock speed of 41.67 MHz.

You can test drive the new PIC18F97J60 by way of the PICDEM.net 2 development board. My personal PICDEM.net 2 is featured in Photo 1 (p. 66). From what I can tell from the schematics, everything on the PICDEM.net 2 is geared toward 3.3 VDC. However, if you look closely, you'll see a dual 5- to 3.3-VDC power supply setup. I stuck a meter across the LCD pins and, just as I had guessed, the LCD had a 5 VDC requirement. The 5-VDC power rail is also part of the PICDEM.net 2's PICtail interface.

If you're wondering about the set of Bel Fuse MagJack magnetics, there is a method to the madness. The MagJack to the far right is supported by an ENC28J60 in a QFN package. When I got the PICDEM.net 2, there was mention of an application that the Microchip developers are working on that will use the second ENC28J60 Ethernet port. That's all I know right now.

As of this moment, the new PICDEM.net 2 runs the PIC18F97J60-capable 3.60 version of the free Microchip stack. I've tested version 3.60 on my ENC28J60-based Frame Thrower II hardware and everything still works as designed.

WHAT'S THE POINT?

With the advent of the new PIC18F97J60, why even bother with the ENC28J60? The answer is that the ENC28J60 is still a viable way to Ethernet enable a multitude of microcontrollers that can be had from various vendors.

The same idea overflows into the meaning behind my garage stack code. You can't legally copy the Microchip stack for commercial use with non-Microchip microcontrollers. So, to implement that special Ethernet project using ENC28J60 technology, you must understand how it works and how best to apply it. My garage stack is currently at version 2.0. I've included the current version and the enhanced DMA-laced code as a separate file for download from the *Circuit Cellar* FTP site.

Before I sign off, I would like to extend my thanks to the Microchip staffers that sat in on my MASTERS classes. They provided the class with insight and answers concerning various facets of ENC28J60 operation that were beyond my reach. With the leap into single-IC Ethernet, the PIC18F97J60 has proven that it really doesn't have to be complicated to be embedded.

Fred Eady (fred@edtp.com) has more than 20 years of experience as a systems engineer. He has worked with computers and communication systems large and small, simple and complex. His forte is embedded-systems design and communications.

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2006/196.

SOURCES

Frame Thrower II
EDTP Electronics, Inc.
www.edtp.com

PIC18F97J60 Microcontroller, PICDEM.NET 2, and ENC28J60 Ethernet controller
Microchip Technology, Inc.
www.microchip.com

BitScope
Pocket Analyzer

USB Oscilloscope & Logic Analyzer

8 Channel 40MS/s Logic Analyzer
Capture digital signals down to 25nS with arbitrary trigger patterns.

3 Input 100MHz Analog DSO
Classic Analog Scope using a standard x1/x10 BNC probe. Additional inputs on the POD for dual channel operation.

8 + 1 Mixed Signal Scope
True MSO to capture an analog waveform time-synchronized with an 8 channel logic pattern triggered from any source.

Real-Time Spectrum Analyzer
See the spectrum and waveform of analog signals simultaneously and in real-time

Waveform Generator
Load up to 32K arbitrary waveform and replay via the onboard DAC (10MS/s) or a digital pattern from the POD (40MS/s)

The new generation Scope for the age of microelectronics.



Turn your PC or NoteBook into a powerful Scope and Logic Analyzer!

See inside your circuit in the analog and digital domains at the same time to make tracking down those elusive real-time bugs much easier.

Pocket Analyzer combines a high speed sample-synchronized storage scope and logic analyzer with a programmable waveform and logic pattern generator. Also included is an integrated real-time spectrum analyzer and powered "Smart POD" expansion interface so you've got all bases covered!

About the same size and weight as a Pocket PC, this USB powered BitScope needs no bulky accessories. It's the perfect low cost "go anywhere" test and debug solution.





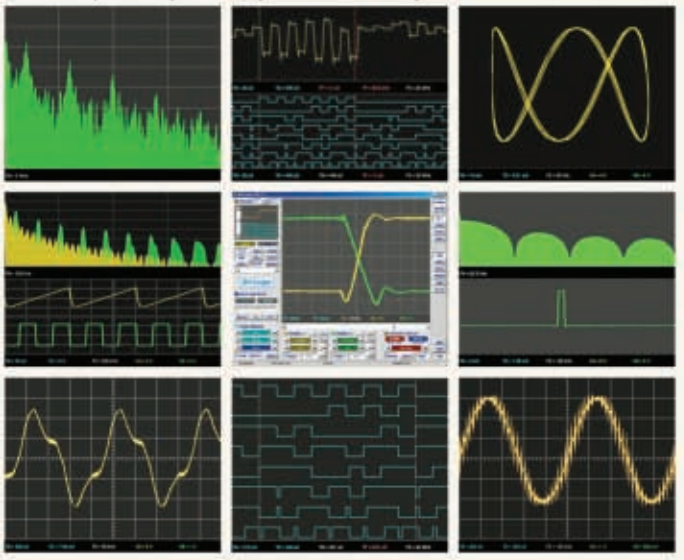
Standard 1M/20pF BNC Input
200uV-20V/div with x10 probe
S/W select AC/DC coupling
S/W select 50ohm termination
Arbitrary Waveform Generator

BitScope "Smart POD" Connector
8 logic channels, 2 analog channels
Dual channel capture from POD A/B
Async serial I/O for external control
Logic Pattern generator 32K 40MS/s

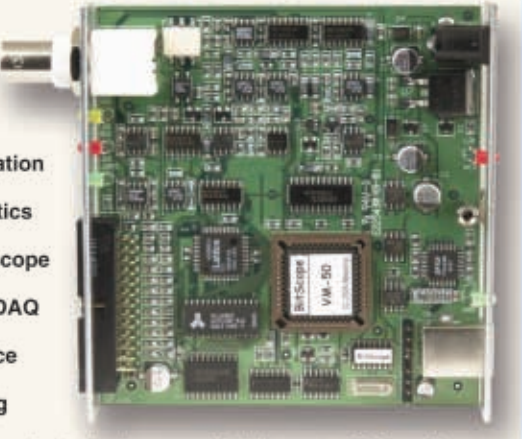
BUS Powered USB 2.0 Device
Single USB cable to your PC
Compressed data transmission
Simple ASCII control protocol
BitScope Scripting Language

External/Passthru Power Supply
Auto senses an external supply - removes power load from USB for use with unpowered hubs.
Supplies up to 500mA via POD

BitScope and your PC provide an array of Virtual Instruments



- R&D
- Education
- Robotics
- Lab Scope
- Fast DAQ
- Service
- Debug



BitScope Pocket Analyzer uses highly integrated Surface Mount technology to provide functionality you would expect from scopes many times the size and price. Its programmable Virtual Machine architecture means new functionality can be added via software. For custom Data Acquisition, export directly to your spreadsheet.

BitScope DSO 1.2 software for Windows and Linux

www.bitscope.com



by Jeff Bachiochi

Smart Character Display

MCU-Based Information Messaging Made Simple

Back in 1990, Jeff built a simple microcontroller-based LED moving message display. Today you can build a "smarter" PIC16F737-based system for a lot less money.

Most of the advertisements (or simply information) thrown at us are static. Billboards and signage have the same look yesterday as they do today and will have tomorrow. But we're seeing the landscape change. We can observe dynamic displays on buses, at ballparks, on gas pumps, on movie theater marquees, and in a growing number of other areas. Although many are used solely for informational purposes, you can't help but feel the increasing rate of exposure to promotional material as well.

Back in 1990, I wrote about building an LED moving message display ("Building an LED Moving Message Display," *Circuit Cellar* 13). For that project, I used latches to hold row information on eight 5 × 7 matrix LED displays. A microcontroller cycled through the five columns of each display continuously writing data.

Today, thanks to the decreasing costs of microcontrollers, it is possible to simplify the hardware necessary to create a smart display. High-current latches can be replaced by many of today's microcontrollers because they can supply milliamps of LED currents directly (usually at a lower cost).

LCDs

Although most character LCDs have a parallel connection, I've found it very useful to be able to add an LCD to a serial output. Because a parallel interface requires from 6 bits (minimum) to as much as 11 bits of I/O, you can easily see that unless your project has ample extra I/O, the parallel interface is impractical. It is almost always possible to find one output that can be used to pump out serial data. So, as a user-friendly peripheral display device, it makes sense to use a serial output TTL interface for this project. (Of course, additional circuitry can be used to convert to almost any kind of interface.)

SMART CHARACTER DISPLAY

The standard character set of most LCDs is a 5 × 7 (row) matrix with an eighth row that's sometimes used for underling and descenders. I've based this design on a single character using a 5 × 8 dot matrix display (see Photo 1). I want the circuitry for each character to be contained within the displays boundaries. In this case, the dimensions are 2.4″ by 1.5″.

A smart character display (SCD) will therefore consist of a single circuit that can be configured alone or in groups of any number of characters or rows totaling 128 characters. A 7-bit address jumper configures each smart display.

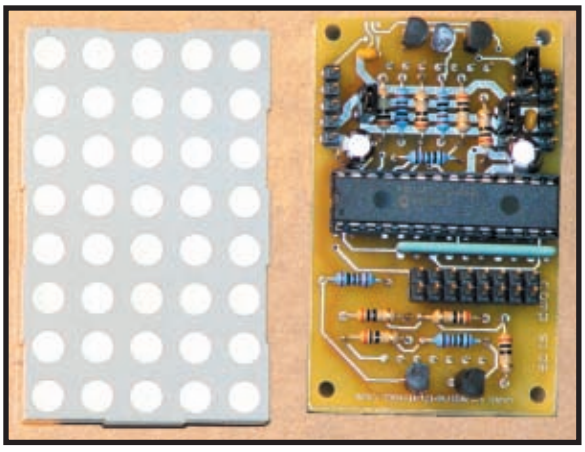


Photo 1—*The smart display circuitry is on a PCB that's identical in size to the LED dot-matrix display. Multiple modules can be stacked to make longer rows and columns to create signage of various dimensions.*

The 7-bit ID indicates the character position in which it will reside. It will only pay attention to the data at its 7-bit ID position. Although each character to display is a single ASCII byte, the data needed to actually display the ASCII character consists of five 8-bit bytes (1-bit for each dot in the row of each column). The display data bytes are found in a table of 5 bytes for each ASCII character.

Like its LCD counterpart, the smart display takes care of multiplexing its own LED matrix. The data remains until the character position is rewritten by an ASCII character or some control sequence (i.e., clear screen).

DUMBNESS

If all displays merely listen to the serial input and handle only the display of their character (ID number), I consider this a dumb smart display. If the first display listens to the serial input and alters the serial input to the rest of the displays, I consider this a truly smart display. What's the difference? This will be discussed a bit later after the basics are understood. Let's start with what's necessary for the simplest operation: Dumb mode.

Any printable characters (0x20–0x7F) received will simply be placed into the display buffer. The buffer is 128 characters long. The buffer's pointer will merely roll over on the one hundred twenty-ninth character and again begin placing characters at the beginning of the buffer. Characters (0x00–0x1F) that do not have a specific function will be tossed out. A few of these control characters will

We make chips think

HI-TECH Software delivers the industry's most reliable embedded software development tools and compilers for over 12 different 8-bit, 16-bit, 32-bit, and DSP chip architectures!



As one of the top five compiler vendors with the largest variety of supported chip architectures, HI-TECH Software's product range is renowned for delivering cutting-edge technology and robust results for development teams worldwide.

With over two decades of industry experience, our long-term relationships with leading chip manufacturers ensure that our products are tightly attuned to new technological releases.

Whichever processor family you are targeting, HI-TECH Software's C compilers can help you write better code and bring it to market faster.

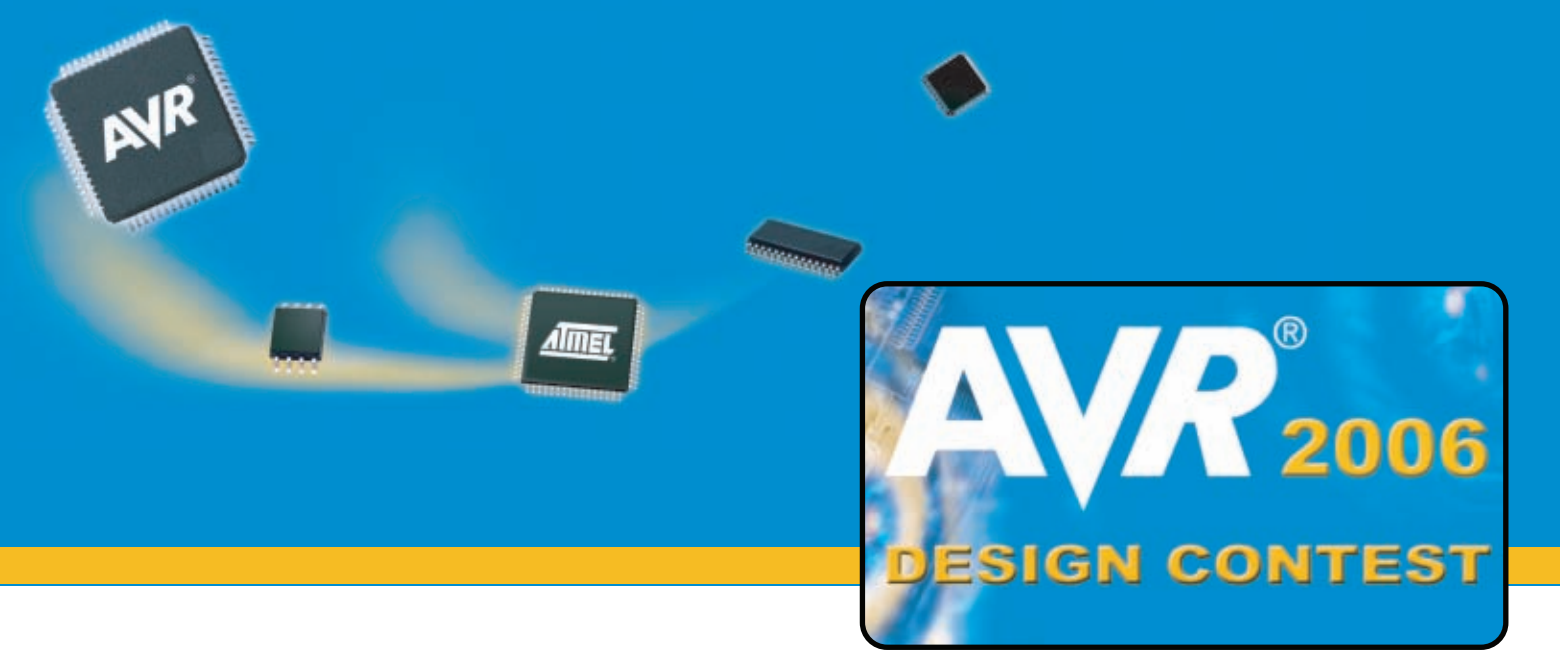
- HI-TECH PICC™ Enterprise Edition
- HI-TECH PICC™
- HI-TECH PICC-18™
- HI-TECH dsPICC™
- HI-TECH C® for ARM®
- HI-TECH C® for 8051
- HI-TECH C® for MSP430
- HI-TECH C® for HOLTEK MCU
- HI-TECH C® for ARClite™
- HI-TECH C® for XA
- HI-TECH C® for Z80
- HI-TECH C® for H8/300
- HI-TECH C® for 68HC11

To see how our compilers can improve your productivity, download a demo now at *www.htsoft.com/downloads/demos.php*.

HI-TECH C® is a registered trademark of HI-TECH Software. HI-TECH PICC™, HI-TECH PICC-18™ and HI-TECH dsPICC™ are licensed exclusively to HI-TECH Software by Microchip Technology Inc. All other trademarks and registered trademarks are the property of their respective owners.

HI-TECH Software LLC 6600 Silacci Way Gilroy, CA 95020 USA
Ph: 800 735 5715 Web: http://www.htsoft.com/

HI-TECH SOFTWARE

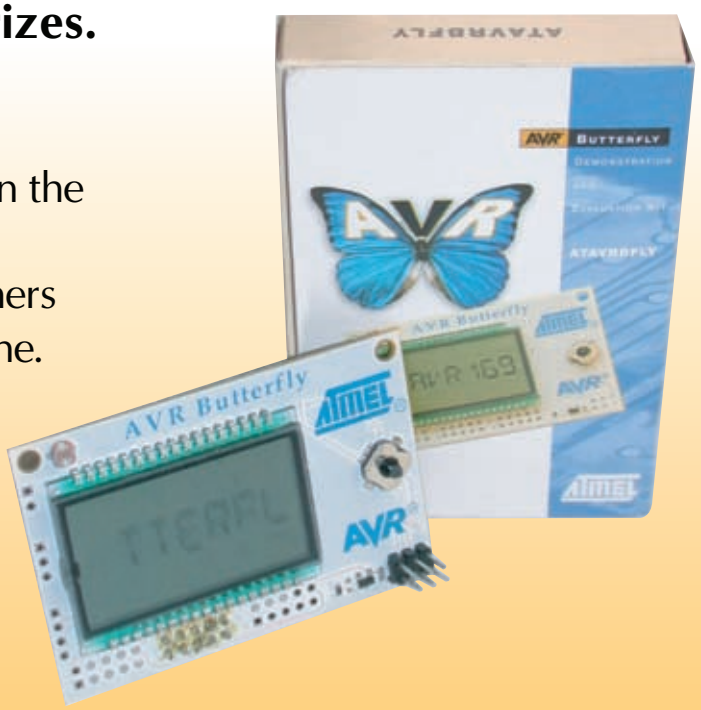


The Atmel AVR Design Contest 2006

is your vehicle to fortune and fame!

$15,000 in cash prizes.

Thank you to everyone who participated in the AVR design contest. The judges are now reviewing the contest entries and the winners will be announced in Circuit Cellar magazine.



For more information about this contest, visit www.circuitcellar.com/avr2006

CIRCUIT CELLAR®
www.circuitcellar.com

ATMEL®
www.atmel.com

perform specific tasks (see Table 1).

Individual character displays contain the entire serial buffer. Receiving serial data is the primary function of each smart display. This serial receive interrupt determines whether a received character is a control byte or display data, and it performs either a control task or moves the data into the character buffer. I limited the table to 7-bit ASCII characters from 0x20 to 0x7F. ASCII characters (from 0x20 to 0x7F) are simply placed in the display buffer at the present buffer pointer position (and the pointer is advanced). Characters received above 0x7F are treated as space characters (0x20) and placed in the display buffer. (The table could certainly be expanded for special characters.) Characters below 0x20 go through the control filter to determine if any special processing must be done.

Control character (ASCII value)	Function
Ctrl-D (0x04)	Offset character pointer = next character (0x00–0x7F)
	Offset column pointer = 0
Ctrl-E (0x05)	Offset character pointer = next character (0x00–0x7F)
	Offset column pointer = 1
Ctrl-F (0x06)	Offset character pointer = next character (0x00–0x7F)
	Offset column pointer = 2
Ctrl-G (0x07)	Offset character pointer = next character (0x00–0x7F)
	Offset column pointer = 3
Ctrl-H (0x08)	Offset character pointer = next character (0x00–0x7F)
	Offset column pointer = 4
Ctrl-L (0x0C)	Clear screen
Ctrl-M (0x0D)	Buffer pointer = 0
Ctrl-N (0x0E)	Display on
Ctrl-O (0x0F)	Display off
Ctrl-P (0x10)	Buffer pointer = next character (0x00–0x7F)

Table 1—*A few control characters enable some simple control over the display including the placement of characters. If multiple rows of text are desired, the buffer pointer can be used to position text anywhere in the 128-character buffer.*

This project handles a few control processes. These processes include clearing, enabling, and disabling the character display (display buffer) and changing the value of the buffer pointer in some way. Character 0x0C (Ctrl-M or carriage return) will clear the buffer pointer. This will direct the next character into the first position of the display buffer and replace the character that is already there. Character 0x0B (Ctrl-L or form feed) will fill the buffer with all spaces (0x20) and clear the buffer pointer. Receiving the control characters 0x04 (Ctrl-D), 0x05 (Ctrl-E), 0x06 (Ctrl-F), 0x07 (Ctrl-G), 0x08 (Ctrl-H), and 0x10 (Ctrl-L) are special cases. These require an additional character. The following received character will be placed into special pointers.

In most cases, Ctrl-P will be used to position the display buffer pointer. If this second character is zero, then this process acts as a Ctrl-M (positioning the pointer to the start of the buffer). Other values position the buffer pointer elsewhere. This enables selective positioning for future printable characters (a second row for instance). Such flexibility lets you divide the linear display buffer into multiple sections.

Ctrl-D through Ctrl-H are macro and micro processes. Although they are meant for the creation of a smart display, they are recognized by each character display and affect macro and micro positioning. Let's see how a character display finds its display data and then focus on the macro and micro positioning.

DISPLAY BUFFER

Each smart character display contains a 128-byte text buffer. This is a ring buffer, and the buffer pointer is limited to 0x00–0x7F. If the eighth bit is always cleared on the buffer pointer, it will automatically roll over. This is the pointer that is affected by a Ctrl-L, Ctrl-M, or a Ctrl-P reception. Any non-control characters that are received will be placed into the text buffer at the buffer's pointer. This pointer is incremented after a character is written to the buffer (see Table 2).

The jumpers on the smart display board are used to select the character the circuit will display in its buffer. This provides you with an extremely flexible system because each smart character display board can display the same or different characters within the message buffer.

CHARACTER TABLE

To determine the actual data the display needs for each column in its 5 × 8 display matrix, the ASCII value of the text character is used as an offset into a table of data. There are 5 data bytes for each ASCII character (see Table 3, p. 76). These bytes (or any) that are saved in five variables (DATA1:DATA5) will be used to actually control the individual LEDs in the display matrix by an interrupt routine.

So, here we are. Any data stored in DATA1:DATA5 will be reflected on the display, so it makes sense that this is an important area of interest. This is where Ctrl-D through Ctrl-H can come into play. Ordinarily, the OffsetCharacterPointer and the OffsetColumnPointer will be zero. And as such, everything works as I've described. The OffsetCharacterPointer is a macro offset that's added to the selected jumper setting to trick the smart character display into displaying a different position within the text buffer. If the OffsetCharacterPointer is 0x01, then each display will actually display the next character position. If the OffsetCharacterPointer is 0x7F, then each display will actually display the previous character position. The OffsetCharacterPointer can shift the portion of the buffer displayed by any number of character positions. This can be used to shift the display between different messages within the buffer. However, if continually changed, it can be used to scroll messages.

Scrolling a message by character position can be a bit jumpy because the characters actually move five columns at a time. For smooth scrolling, you need to be able to move characters by the column, although the

Buffer position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	…	125	126	127
Text in buffer	T	h	i	s		i	s		a		T	e	s	t	!		…			

Table 2—*Each smart character display has a 128-byte text buffer. All displays have an identical buffer. Jumpers on the display board select the buffer position to be displayed. The text character at that position determines an offset into a table to determine what data will be actually displayed from the appropriate character.*

best approach here might be to have a buffer that holds all of the display data (five times the size of the text buffer and five columns of data for each text character). So, for a 128-byte text buffer, you would actually need a 640-byte data buffer. This alone would almost double the cost of the microcontroller.

Another possibility is to keep only the character of interest and not the entire text message in a buffer. This defeats the scrolling ability without having to continuously send the complete text buffer. So, the OffsetColumnPointer is implemented to allow micro positioning of characters within the 5 × 7 matrix.

For a standard non-scrolling message, it is sufficient to do a single table look-up to get all of the data necessary for a single ASCII character. To enable micro positioning, you need to do separate table look-ups for each column. This is because a single smart character display may be asked to display portions of two consecutive text characters.

Micro positioning is an offset that is used in table look-up to point to a particular column's data. The OffsetCharacterPointer value (from 0x00 to 0x04) is added to the column of interest to offset into specific column data within the character table look-up. When this value reaches the last of the five data elements in the table, a flag is set to increment the character position if more column data is needed. If so, this data will come from the next text character (a different table entry).

Five control characters are used for macro/micro positioning. Ctrl-D through Ctrl-H each require an additional value to be sent. This value is placed in the OffsetCharacterPointer. The OffsetColumnPointer is determined by the control character (Ctrl-D = 0, Ctrl-E = 1, etc.). In this way, both the OffsetCharacterPointer and the OffsetColumnPointer are changed together to prevent bad data.

		Entry offset 0	Entry offset 1	Entry offset 2	Entry offset 3	Entry offset 4
Table Entry 0x20 (space)	Bit7	1	1	1	1	1
	Bit6	1	1	1	1	1
	Bit5	1	1	1	1	1
	Bit4	1	1	1	1	1
	Bit3	1	1	1	1	1
	Bit2	1	1	1	1	1
	Bit1	1	1	1	1	1
	Bit0	1	1	1	1	1
Table Entry 0x21 (!)	Bit7	1	1	0	1	1
	Bit6	1	1	0	1	1
	Bit5	1	1	0	1	1
	Bit4	1	1	0	1	1
	Bit3	1	1	1	1	1
	Bit2	1	1	1	1	1
	Bit1	1	1	0	1	1
	Bit0	1	1	1	1	1
Table Entry 0x22 (")	Bit7	1	0	1	0	1
	Bit6	1	0	1	0	1
	Bit5	1	0	1	0	1
	Bit4	1	1	1	1	1
	Bit3	1	1	1	1	1
	Bit2	1	1	1	1	1
	Bit1	1	1	1	1	1
	Bit0	1	1	1	1	1
Table Entry 0x23 (#)	Bit7	1	0	1	0	1
	Bit6	1	0	1	0	1
	Bit5	0	0	0	0	0
	Bit4	1	0	1	0	1
	Bit3	0	0	0	0	0
	Bit2	1	0	1	0	1
	Bit1	1	0	1	0	1
	Bit0	1	1	1	1	1

Table 3—*These are the table entries for the first four ASCII characters 0x20–0x23. Each character has 5 bytes of data, one byte for each column in the 5 × 7 matrix. I listed each entry by bits so you can visualize how the data itself forms characters. The highlighted 0 data will turn on associated LEDs.*

Timer0 INTERRUPT

Each of the microcontroller's I/O pins can source/sink approximately 25 mA (with a total device current not to exceed 250 mA). Each row of the LED matrix is connected to an output bit on PORTB. Any data "0" written to a PORTB's output bit will sink that row to ground (allowing LED current). One port is used to output data, and each of the five columns of LEDs will need different data. Each column must provide LED current only when the associated data for that column is available. If all of the LEDs in a single column are to be illuminated, the total current necessary is eight times the current of a single LED. A microcontroller's I/O pin cannot handle this high current. Therefore, each of the five columns has a PNP transistor to supply that current when necessary. PORTA supplies the control for each of the transistors (see Figure 1). Column control on PORTA must be synchronized with the appropriate data placed on PORTB. This is controlled by the Timer0 overflow interrupt routine.

The clock for Timer0 is derived from the internal oscillator—which is set to 8 MHz in this case—divided by four (0.5 µs). Note that this is also the execution time for one instruction cycle. Timer0 is an 8-bit timer that feeds through a prescaler. The prescaler is set to divide by 16, giving a timer tick of 8 µs. Therefore, Timer0 will roll over once every 2 ms (8 µs × 256). At each interrupt, a column will have its PORTB data updated and its column transistor enabled. This column will be enabled until the next interrupt. All five columns are updated in five interrupts or 10 ms (100 Hz), which is the refresh rate. This is fast enough to fool your eyes into believing that all of the illuminating LEDs are on at the same time.

MATRIX LED MODULES

There are various kinds of matrix LED

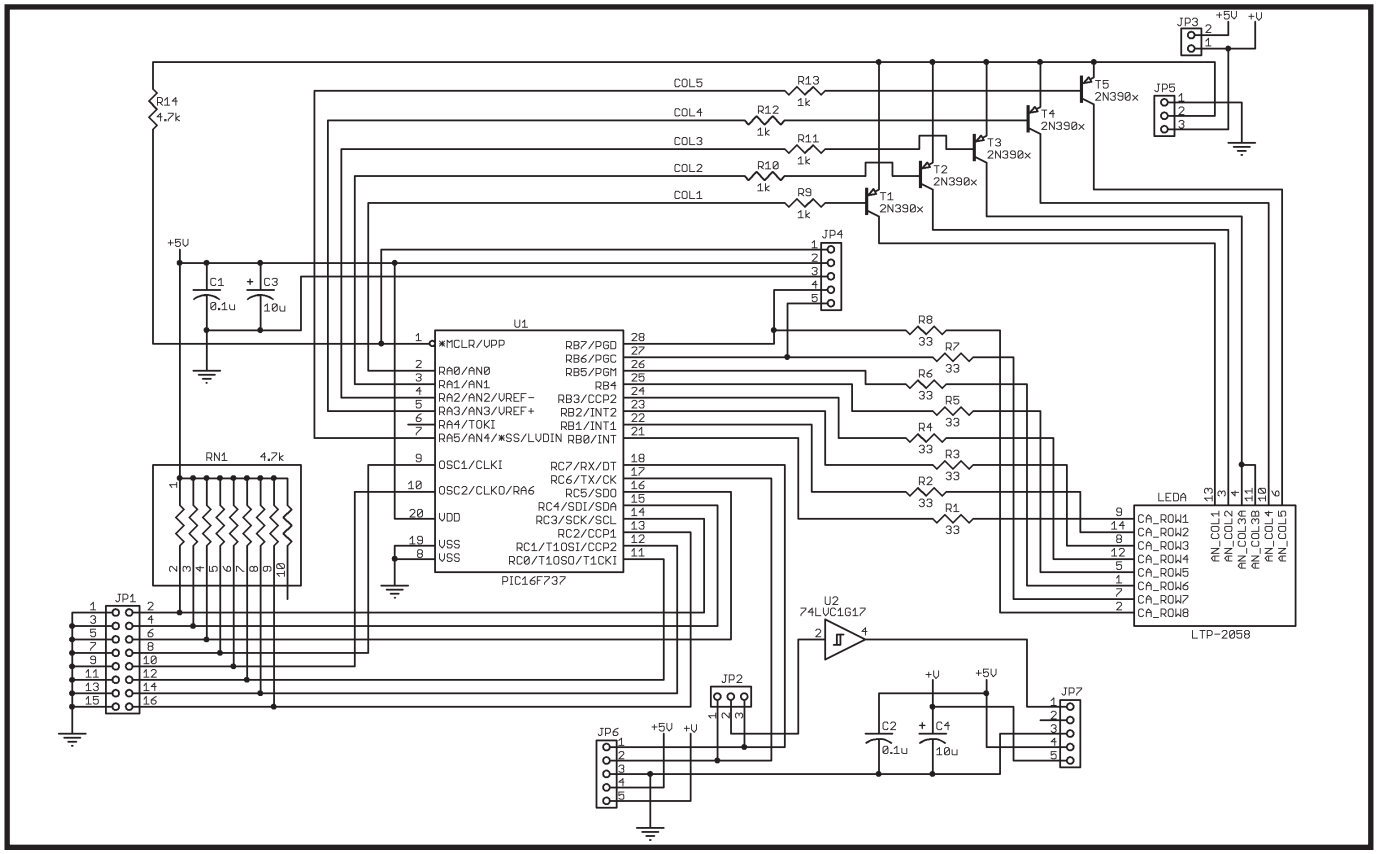


Figure 1—*The smart display consists of a small Microchip Technology PIC16F737 microcontroller that handles communications and multiplexing for each 5 × 7 dot-matrix character display. Both common anode and cathode column displays can be used with a change to the column driver transistors. Although the system and LED 5-V power sources are separated, they can be jumpered together when a small number of characters is used (currents are low).*

modules: common anode column/common cathode row and common cathode column/common anode row. The difference is the polarity of the individual LEDs to the common columns and rows. This project uses common anode column/common cathode row modules sourcing current through PNP column transistors connected to 5 V. Common cathode column/common anode row modules can be used by sinking current through NPN transistors connected to ground. A single jumper (JP5) is used to select the proper connection for either module. This jumper is sampled via an input pin, and it compliments all of the data when necessary to automatically support either type of module.

SMART/DUMB

As you can see in Figure 1, the serial data is buffered to prevent loading on the serial connection. The buffer's data comes from either the main serial connection or from the microcontroller's serial output. This allows the first smart character display to become a smart master to the rest of the connected dumb slaves. When configured as a master, the first SCD can pass through all communications from the main serial connection and continuously add the necessary control commands to allow for automatic scrolling. There are in fact many ways this can be used. You might want to automatically cycle through multiple messages. You can also assemble a panel of characters and treat the display as a graphics interface.

THROUGH HOLE VERSUS SMT

My prototypes were made with through-hole technology. However, I am in the process of laying this out for SMT as well. You'd think that using smaller parts would just make the job easier. I started by placing SMT LEDs using the same spacing as my LED modules. This quickly demonstrated the advantage of using the through-hole LED module. Row and column connections severely limit the available routing real estate left for the remaining circuitry. The cost of using an LED module (even in single quantities) works out to about $0.15 per LED. Some of the cheapest SMT LEDs are not any less expensive (and these have lower intensity). So, the through-hole matrix LED modules may still be the best choice even if the remaining components are surface mount!

Jeff Bachiochi (pronounced BAH-key-AH-key) has been writing for Circuit Cellar *since 1988. His background includes product design and manufacturing. He may be reached at jeff.bachiochi@circuitcellar.com.*

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2006/196.

SOURCES

LTP-2158/LTP-2058 LED Module
Lite-On Technology Corp.
www.liteon.com

PIC16F737 Microcontroller
Microchip Technology, Inc.
www.microchip.com



Memory Lane Change

Get ready for a walk down memory lane. Tom describes the major paradigm shifts that have occurred in memory technology over the past three decades and then readies you for what's ahead.

Nowhere is the march of silicon demonstrated so clearly than with memory chips, where it all began some three decades ago.

Everybody knows Intel created the 'x86 (8088) microprocessor that powered the original IBM PC. But do you remember that they got their start years earlier with the 1103 dynamic RAM (DRAM) chip? Simply put, no DRAM, no PC, or any of the other digital A/V gadgets (e.g., video game and HDTV) we take for granted.

I certainly don't design PCs, video games, or HDTVs, and I doubt many of you do either. Thus, to tell the truth, I haven't kept up with developments on the DRAM front. Time to, pardon the pun, refresh my memory.

Let's see, the latest and greatest DRAMs are 2-Gb chips that run from 500 to 800 MHz.[1] That's roughly 2 million times the density and 200 times the speed of the 1-Kb, 300-ns access time 1103. You don't need a PC crunching those numbers to understand that a 400 million times improvement is a big deal. And that's not even considering issues like chip size, reliability, and power. (The 1103 needed 19- and 22-V supplies!) What the heck, let's just say modern DRAMs are a billion times better than the original. If you feel that's an exaggeration, patience, grasshopper. As with all things silicon, it's only a matter of when, not if.

The invention of the DRAM was a true example of a "paradigm shift" unlike most (ab)uses of that overworked phase. Almost overnight, the big computer companies of the era Dr. Jacked core memory in favor of DRAM (see Photo 1). It wasn't more than a couple of years after the 1103's introduction that it was the single biggest selling chip in the world.

Lest you think Intel's 1103 was a fluke, keep in mind that their memory credentials also include big winners like EPROM and the flash memory that replaced it (another paradigm shift). On the other hand, the ride down memory lane was not without some potholes. (Remember "Bubble Memory"?)

It's a bit ironic that core memory had features (i.e., the accessibility of RAM and nonvolatility of flash memory) that still elude mainstream memory technologies. But maybe that's about to change. Are there memory technologies on the horizon that could obsolete DRAM and flash memory just as they did their predecessors? Paradigm shift anyone?

LIGHT MAKES RIGHT

Given the history lesson above, what better place to start our tour down memory lane than at Intel? Although their 'x86 machinations may get all the headlines, behind the scenes, they're hard at work searching for the next big thing memory-wise.

For now, the current memory technologies (e.g., DRAM and flash memory) are doing just fine. But the chip wizards are saying it won't be long before the party is over for these historic designs as challenges to continued scaling (aka "the wall") loom ever closer.

Stefan Lai's "Non-Volatile Memories: A Look Into the Future" puts the situation in stark perspective. According to Intel, the good news is that flash memory has gone through eight process migrations, shrinking cell size by almost a factor of 500 from the 1.8 µm original to the currently shipping 90 nm. The bad news is that there are only a couple of more shrinks in the pipeline (65 nm, which Intel just started sampling, and then 45 nm) before current designs hit "the wall." In other words, the party is 90% over.

That doesn't mean DRAM and flash memory are doomed to join vacuum tubes and core memory on the ash heap of computer history. The wizards may yet have a few tricks up their sleeves. For instance, there's talk of potentially increasing the number of bits per cell from 2 (i.e., a four-level cell) to 4 (i.e., a 16-level cell), although the tiny voltage margins (e.g., 0.1 V) would certainly pose a challenge. Nevertheless, march of silicon naysayers have been proven wrong many times in the past, and I suspect flash memory and DRAM will be around for a long, long time. But it will also take a long time for any successor to get ramped up, so there's no time to dillydally.

Although Intel is reportedly investigating many different approaches, the leader appears to be Ovonics Unified Memory (OUM), which is also known as Phase-Change Memory (PCM). One of the main things going for this new idea is that it isn't a new idea at all. Stanford Ovshinsky of Energy Conversion Devices first discovered the approach in the '60s. In 1970,

engineers from that company contributed to an article that appeared in *Electronics* magazine describing an "Amorphous Semiconductor Memory."[2] The other coauthor was, surprise, one Gordon Moore of then brand new startup Intel.

Ironically, favorable prospects for OUM chip memory owe a debt to advances in rotating memory, namely read-write CDs and DVDs. Thanks to the materials refinement and production know-how gained with optical drives, OUM is now poised to live up to the promise made nearly 50 years ago.

As you may recall, read/write optical drives exploit the property of chalcogenide glasses to change state when heated. In a CD-RW or DVD-RW, a laser is used to provide the heat, and the property of the glass that is exploited is optical, namely the difference in reflectivity between the amorphous (dull) and crystalline (shiny) states.

An OUM uses basically the same material and is conceptually similar, but with two key differences (see Figure 1). First, IC process-formed heating elements replace the optical drive's laser. Second, the state change is measured electrically rather than optically (i.e., resistance is high in the amorphous state and low in the crystalline state).

As far as blue-sky stuff goes, OUM is a relatively real deal. Besides Intel, other heavyweights signed-up include Samsung and STMicroelectronics. Significant test chips that go well beyond the usual dog and pony show prototype have already been produced, such as a 256-Mb unit from Samsung.

Remember in *The Graduate* (1967) when Dustin Hoffman gets the one-word career advice "plastics"? Plastic indeed may be the future if another memory technology Intel is exploring, Polymeric Ferroelectric RAM (PFRAM), pans out. The beauty of this scheme is that the memory cell itself, like OUM, has no transistors (and thus no transistor scaling issues). Furthermore, layers can be stacked for even higher density.

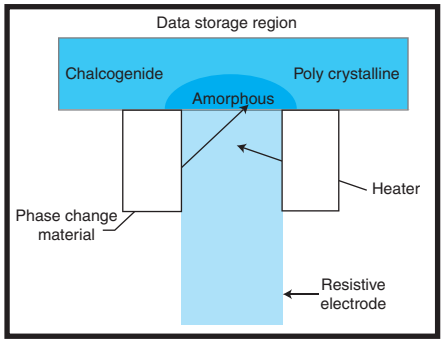


Figure 1—*Look, Ma, no transistors. Ovonics Unified Memory and other transistor-less media technologies are preparing for the day DRAM and flash memory hit "the wall."*

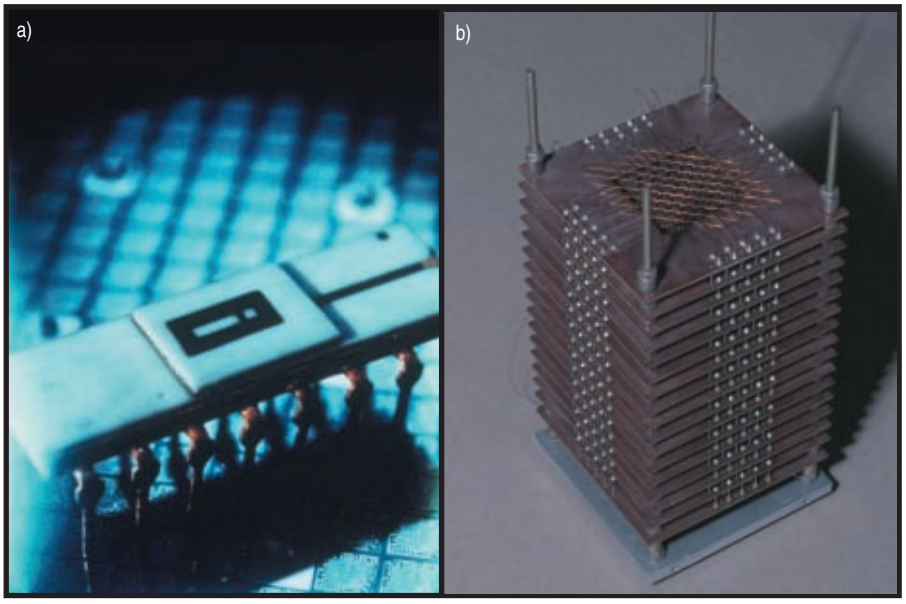


Photo 1—*Almost overnight, DRAM **(a)** killed core memory **(b)**. It's not a matter of if a new memory technology will return the favor, but it's definitely a matter of when. (1103 DRAM photo credit: www.cpu-museum.com/Memories_e.htm. Core photo credit: http://physics.indiana.edu/~meyer1/historinstr/ pics/coremem1/page1.htm.)*

Indeed, the 3-D approach is an intriguingly simple, if admittedly brute-force, approach to cramming more functionality into less space. We're already seeing the benefits in the variety of "stacked die" parts appearing on the market. Matrix Semiconductor is an outfit that takes the concept one step further. They whimsically observe that silicon wafer costs are now approaching $1 billion per acre. So, instead of stacking die, a better approach is to make "high-rise" chips that stack layers of memory on a single die (see Photo 2). The only downside is that their anti-fuse-based chips are only one-time programmable (OTP). Nevertheless, Matrix is on a roll, as evidenced by their recent acquisition by SanDisk (for a cool $250 million) and a list of investors that includes heavyweights like Sony, Nintendo, Seagate, and Kodak. Note that the OTP limitation could actually be a "feature" for certain applications such as piracy-proof prerecorded media or "black box" data loggers.

Like current memory chips, all of the approaches discussed so far are so-called "XY" addressable. That means lithography limits remain a challenge for accessing the memory elements even if the elements themselves are no longer lithography-bound (i.e., they aren't transistor based).

Once again, taking a page from the rotating memory playbook, so-called "seek-and-scan" memories separate the read/write and access function in much the same way as disk drives have a read/write head "seeking and

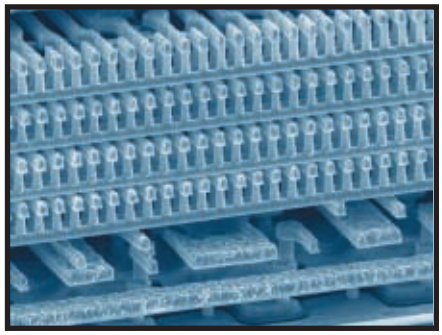


Photo 2—*According to Matrix Semiconductor, which was recently acquired by SanDisk, when the real estate gets expensive, it's time to build up as shown here with one of their "four-story" chips. They cost less than flash memory, but they're only one-time programmable, which may be an acceptable or even desirable trade-off in certain applications.*

PC/104 Single Board Computers

Low Price, Low Power, High Reliability using Linux development tools



TS-7200
shown with
optional A/D converter,
Compact Flash and RS-485

- options include:
  onboard temperature sensor, A/D Converter 8 channel 12 bit, Extended Temperature, Battery Backed Real Time Clock, USB Flash 256 M (with ARM Tool Chain), USB WiFi

200 MHz ARM9
Power as low as 1/4 Watt

- 5 boards, over 2000 configurations — $99 qty 100
- Fanless, no heat sink
- SDRAM - up to 128M — $129 qty 1
- Flash - up to 128M onboard
- 10/100 Ethernet - up to 2
- DIO lines - up to 55
- 2 USB ports
- COM ports- up to 10
- Programmable FPGAs
- Linux, Real Time extension, NetBSD

NEW!
- SD card option
- VGA video

Off-the-Shelf Solutions ready to design into your project using DOS development tools



TS-5600 Shown with
optional flash modules,
A/D, RS-485 and
Merlin cellular modem

- options include:
  RS-485 Half and Full Duplex, A/D Converter up to 8 Channels at 12 bits, DAC up to 2 Channels at 12 bits, Extended Temperature

133 MHz 586

- Power as low as 800mA — $229 qty 1
- Fanless, no heat sink
- SDRAM - up to 64M — $259 qty 1
- COM Ports - up to 4 ports
- Ethernet Ports
- DIO Channels - up to 40
- PCMCIA II adaptor
- Compact Flash adaptor
- USB Ports (Except on TS-5300)

see our website for 33 MHz 386 configurations

- Over 20 years in business
- Open Source Vision
- Never discontinued a product
- Engineers on Tech Support
- Custom configurations and designs w/ excellent pricing and turn-around time
- Most products stocked and available for next day shipping

Design your solution with one of our engineers (480) 837-5200

New Products and PC/104 Peripherals

Tiny WiFi Controller boots Linux in 1.6 seconds



NEW!

$249 qty 1

- 200 MHz ARM9
- Up to 128MB Flash
- Up to 128M SDRAM
- 802.11g WiFi
- SD Flash Card socket
- 1 external USB port
- 1 10/100 Ethernet
- 3 TTL serial ports
- Rugged aluminum enclosure measures 1.1" x 4.9" x 3.1"

Intelligent Battery Back-up

NEW!

$119 qty 1

- Run your system for days with no external power source



Modems	33.6K baud, 56K baud, AT commands, caller ID, cellular using GSM and CDMA technologies
Non-volatile Memory	up to 2MB, 10 year lithium battery
Serial Ports	up to 4 serial ports with optional RS-485
Opto-isolated Serial Port	opto-isolated RS-485 on 2 ports
12 bit A/D, DAC	8 channel 12-bit A/D converter, optional 2 channel 12-bit DAC, A/D jumpered for 0-2.5V, 0-10V or 0-20mA
CAN Bus Controller	Philips SJA1000, opto-isolated, up to 1 megabit/sec selectable termination resistor, Ocera Linux driver
NEW! 64 Digital I/O	32 inputs, 32 outputs, 200 mA drive, optional 512 Kbyte or 1 MB battery-backed SRAM, stack up to four boards, RoHS compliant

see our website for more boards and option details

Technologic SYSTEMS

We use our stuff.

Visit our TS-7200 powered website at

www.embeddedARM.com



(Check out that Jameco Smile!)

Big savings.

10 to 20% off — on all of the industry's name-brand components. Every day. Across the board.



Or even bigger savings.

20 to 30% off — on top-quality ValuePro and ReliaPro components. Totally guaranteed by Jameco.



Jameco gives me more choices and smarter ways to save!

Big names at best prices
For parts like memory and microprocessors you'll likely want name brands...

So Jameco gives you all of those at the lowest prices in the industry. *Guaranteed in writing!*

But that's just the start.

Save even more— whenever you want to...
In passives, interconnects, electromechanicals, power products, motors and similar parts, Jameco offers you a unique *additional* way to save...

In addition to name brands you can also choose from *ValuePro* and *ReliaPro* brands—and cut *20 to 30%* off of name-brand pricing!

Actual Jameco Savings vs. Other Major Catalogs

	Name Brands	House Brands
LED, T-1¾, Red, 1200mcd LEDTech LT1873-81-UR-P22	12.5–44.0% Savings	20.8–49.3% Savings
Connector, RF, BNC, 50Ω Amphenol 31-202	17.1–27.9% Savings	31.6–40.5% Savings
Cable/Wire, CAT 5E Belden 1000'	10–20.6% Savings	55–60.3% Savings
Molex Mini-Fit Jr™ R/A Gold Molex 39-30-1241, 24 pos.	14.7–37.3% Savings	22.1–70.7% Savings
Trimmer Pot, ¼" rnd, 1kΩ, .5W Bourns 3329P-1-102	15.6–28.9% Savings	24.6–36.5% Savings

See complete details at www.jameco.com/comparisons.



And all of these parts are Grade-A quality. Fully guaranteed by Jameco.

(Fact is, they're often made by the same manufacturers who supply the name brands. But Jameco buys direct and saves you all the middleman cuts!)

More choices, top savings
Name brands or house brands, you always get more ways to save at Jameco. Call 1-800-831-4242 for a free catalog. Or go direct to the website...





JAMECO® ELECTRONICS

Great Products. Awesome Prices. You'll Be Smiling Too!

So what are you waiting for? See for yourself, right now, at www.Jameco.com/cct

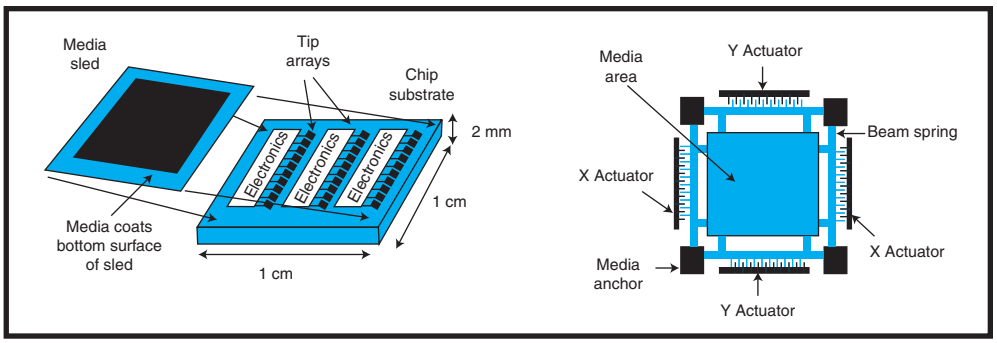


Figure 2—*A disk drive is certainly an electromechanical structure, so why not use MEMS technology to micro-size it? MEMS disk drives use a "media sled" and array of probes to emulate the rotating platters and seeking read/write heads of a disk drive.*

scanning" large platters of media.[3]

Companies like IBM and HP have done pioneering experimentation with seek-and-scan. Another new company, Nanochip, has recently licensed the OUM technology described earlier. Although details are sparse (the company's web site is under construction), by pairing OUM media with MEMS-based seek-and-scan, the company appears to be shooting for something like the IC equivalent of an optical drive. Instead of rotating platters, an OUM "media sled" will be XY positioned using MEMS "springs" (see Figure 2). It's an interesting idea, and it's also interesting to note that Intel is an early round investor in Nanochip.

Presumably, beyond all of these lie various forms of "nano," "molecular," and "atomic" memory. This is kind of fun to think about because there are on the order of 5 sextillion (5×10^{21}) atoms in a drop of water.[4] Considering, for example, that a 500-GB disk drive holds roughly 5×10^{12} bits, that means a drop of bloatware-be-damned water could hold as much data as a billion disk drives!

SUPER RAM

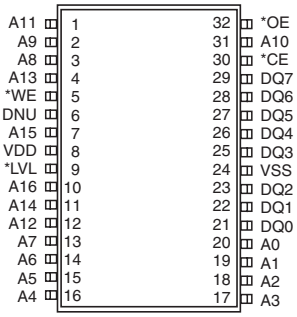
It's a bird! It's a plane! It's SuperRAM!

Er, Earth to Tom: time to get back to reality. OK, how about we take a look at some new-school memories you can buy today (see Figure 3), namely magnetoresistive RAM (MRAM) and ferroelectric RAM (FRAM).

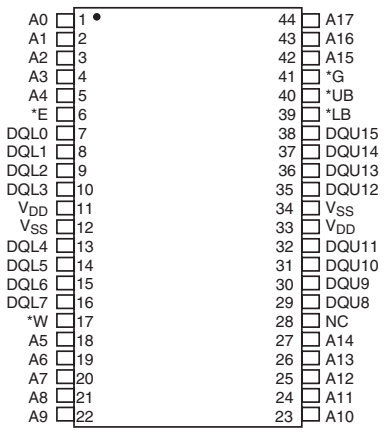
I suppose there's a bit of irony in the fact MRAM works quite like the core memory of yore, with a magnetic tunneling junction transistor standing in for the little magnetic doughnuts. Meanwhile, FRAM uses a special ferroelectric film that is electrically polarized, which kind of reminds me of that nonvolatile LCD I covered a while back ("Batteries Not Included," *Circuit Cellar* 182, September 2005). MRAM and FRAM have advantages over flash memory when it comes to writing in terms of speed, simplicity, and power consumption.

As one of the pioneering FRAM suppliers, Ramtron deserves credit for sticking with it through the tough challenges of pioneering a new technology. I'd say their new 1-Mb part is a good payoff, one that can actually make sense in real applications.

Notably, the Ramtron FM20L08 adopts the classic JEDEC "bytewide" SRAM pinout and functionality. As a point of fact, this means it can target sockets for battery-backed SRAMs, which typically require large high-profile DIP packages to accommodate the battery and can have issues in manufacturing (e.g., cooking the battery). The FM20L08 not only ditches the battery, but also requires less board



Ramtron FM20L08



Freescale MR2A16A

	Technology	Capacity	Access time	Cycle time	VCC	Power-Read	Power-Write	Power-Standby	Endurance	Retention	Package
Ramtron FM20L08	FRAM	1 Mb - 128K × 8	60 ns (25-ns page mode)	150 ns	3.3 V	22 mA	22 mA	20 µA	"Unlimited" (est. > 10^{16} cycles)	10 years	32-pin TSOP (8 × 13.4 mm)
Freescale MR2A16A	MRAM	4 Mb - 256K × 16	35 ns	35 ns	3.3 V	80 mA	155 mA	12 mA	Unlimited	10 years	44-pin TSOP (10 × 18.3 mm)

Figure 3—*Although not dense or cheap enough to challenge DRAM and flash memory in mainstream applications, FRAM and MRAM are now viable options for designs that can take advantage of their unique capabilities.*

space and goes through manufacturing just like any other chip.

The FM20L08 does a decent job impersonating an SRAM with the familiar pins (i.e., *CE, *OE, *WE, etc.) working more or less the usual way, although there are a few differences. For an SRAM, the cycle time and access time are usually the same. But the FM20L08 access time is 60 ns, while the cycle time is a more leisurely 150 ns. Performance-oriented applications can take advantage of a DRAM-like page mode to burst up to 8 bytes at 33 MHz. Another embellishment is a low-voltage lockout function to prevent spurious writes during power failure. After all, what good is a nonvolatile memory if the data that gets written into it is erroneous? Finally, there's a version of the chip available that includes a software write protect scheme as an additional way to harden the data against software and hardware glitches.

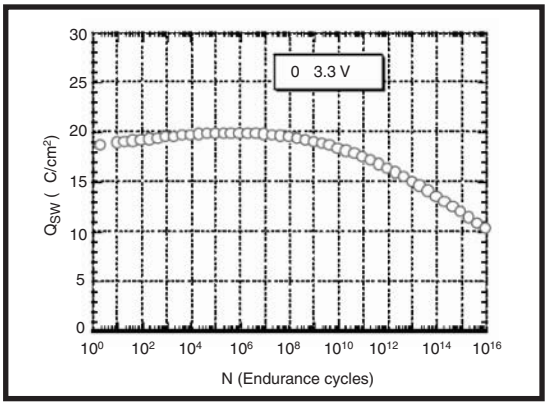


Figure 4—*If not truly a concern in practice, the "destructive read" characteristics of FRAM inspired nagging doubts about endurance for read-mostly-and-often applications. According to Ramtron, those doubts should be laid to rest for their new 0.5-µm, 3.3-V parts. As shown here, they predict that even after 10^{16} cycles, the polarization density of 10 µC/cm² will safely exceed the minimum sensing threshold of 6 µC/cm².*

Meanwhile, Freescale is in volume production with their 4-Mb (256K × 16) MR2A16A MRAM chip. Sharing the FRAM's 3.3-V supply and low-voltage lockout features, the MR2A16A feels even more at home in SRAM sockets thanks to equal—and at 35 ns, fast—access and cycle times. Although it's an *x*16 part, separate upper- and lower-byte select pins allow single byte access.

Although their SRAM-like façade is familiar enough, these parts require a different way of thinking when it comes to minimizing power consumption. For example, the Freescale MRAM standby current spec of 12 mA is so much higher than the microamps of an SRAM that I thought it was a typo. What seems a problem has a simple brute force solution, which is just to cut the power to the chip entirely. Yes, there is a 1-µs delay required after power-up, but that's not a very long time, which means turning off the power for even very short idle periods (e.g., milliseconds) can pay off.

Meanwhile, the FM20L08 features a more typical (i.e., SRAM-like) standby current of 20 µA. On the other hand, its power-up delay spec is much more significant at up to 5 ms. Thus, the "standby" versus "power-down" trade-off is quite different for the FM20L08 FRAM than for the Freescale MRAM.

Meanwhile, active power consumption characteristics for the two parts are also quite different. The 'L08 FRAM gets relatively decent mileage with power consumption of 22 mA (maximum) for both reading and writing. For the MR2A16A MRAM, read power consumption is up to 80 mA (maximum 55 mA typical), while write current is nearly double that at as high as 155 mA (105 mA typical).

However, note that actual system power consumption may differ from that implied by the raw datasheet specs due to the functional differences between the parts, starting with the fact the MRAM has four times the number of bits. In addition, power consumed by the MR2A16A MRAM is potentially reduced because it's twice as fast (35 versus 60 ns access time) and twice as wide. Thus, it theoretically may be less "active" than the FM20L08 FRAM. For example, the MRAM requires only a single 35-ns cycle to move 16 bits, while the FRAM needs two 8-bit cycles to accomplish the same task, which requires at least 100 ns (page mode) or 300 ns (non-page mode).

In either case, it's an improvement over flash memory. Whether handled by a coprocessor or software driver, writing a flash memory chip is a long and complicated process that consumes numerous clock cycles to erase and rewrite entire sectors. And don't forget what the high-voltage (e.g., 15 V) flash memory requires for writing. More instructions, more clocks, more bits, and more volts: it all means more power.

Besides the power consumption itself, flash memory machinations (e.g., busy bit babysitting) are rather an annoying and perhaps untimely chore for the processor. One key concern is what to do if the lights go out midway through the relatively long and unforgiving window of flash memory vulnerability during a sector erase/write sequence. By contrast, to write an MRAM or FRAM you just, well, write it. One instruction, one cycle, one less thing to worry about.

Beyond the Memory 101 basics (speed, power, and price), the Holy Grailness of next-generation memories is all about the nonvolatility. Are these things robust or doomed to silicon senility? On the one hand, early prototypes were definitely prone to a variety of data-disrupting fatigue, stuck bit and write disturb phenomena. Indeed, there are nontrivial challenges like simply figuring out how reliable the memory is short of running a memory test for 10 years.

A notable spec has to do with the "destructive" characteristics of read cycles for FRAM. Like a DRAM, the ferroelectric capacitor-based FRAM requires data to be refreshed (i.e., rewritten) following a read cycle. Although the cycle endurance might be huge, this could prove problematic for applications that are "read mostly and often," such as using an FRAM to store code run by a high-performance processor.

Early FRAMs had endurance specs on the order of 10^8, or 100 million, cycles. The good news is even this first-generation spec was 100 to 1,000 times the right endurance spec for a typical flash memory. But crank the numbers and you can see the potential problem of using FRAM for execute-

in-place (i.e., op-code fetch) applications. Consider that 100-million cycle endurance spec in light of the fact that the FM20L08 could theoretically deliver more than 6 million cycles per second. Even assuming a more leisurely 1 million cycles per second, trouble could be mere minutes away.

Subsequent work to extend FRAM endurance paid off with improved specs on the order of 10^{12}, or 1 trillion, cycles. Now our 1-million-cycles-per-second application can run for 1 million seconds, which sounds good until you realize that is less than 12 days!

But, as represented by the FM20L08, the new generation of FRAMs have made dramatic progress with the latest endurance estimates on the order of 10^{16}![5] Now we're talking something like 12,000 days, or 30-plus years, of 1-million-cycle-per-second operation (see Figure 4).

For MRAM, endurance is less an issue because access (flipping magnetic polarity like a disk drive) has no intrinsic wear-out mechanism. Nevertheless, earlier parts suffered from a "single-line disturb" phenomenon, which is essentially a crosstalk problem in which writing one bit could change others. With their introduction of the MR2A16A, Freescale solved the problem with a clever "toggle" approach. As the name implies, the memory is not set to 1 or 0 directly, but it's read and then toggled if necessary to achieve the desired state. The current waveforms applied on the word and bit lines for the toggle approach are less likely to disturb adjacent bits.

Yes, there may be issues for MRAM and FRAM lurking under the hood that will only become apparent over time. But for now, all you can really do is take the datasheets at their word, including the assertion on the Ramtron FRAM datasheet that it features "unlimited read/write cycles." In terms of data retention, both the Freescale and Ramtron parts specify 10 years. That's a spec I've got application concerns about, although it does not by any means imply that the chips get amnesia after 10 years and a day. Actual environmental and use characteristics have a big impact on retention, and extremes are unlikely in actual practice. Because that 10-year spec is still widely used by flash memory suppliers (although there's a trend to increase it), let's leave the subject aside for now. The bottom line is that the specs for both the Freescale and Ramtron parts say you can treat the thing like a RAM, but it will hold the data like a ROM, at least for 10 years.

THANKS FOR THE MEMORIES

The latest generation of MRAMs and FRAMs work great, but they are by no means the Holy Grail memories that will lay the old to rest overnight. The new chips are just getting over teething pains and simply don't offer the density or cost per bit to displace DRAM or flash memory in mainstream markets. For now, they're niche players. But with that being said, they're still kind of cool.

Stand-alone memory chips are just one part of the story, and frankly the lesser part from my embedded perspective. When it comes to replacing existing memories, I'm just as interested in the memory found on high-integration MCUs and SoCs (notably including FPGAs) that package everything in a single chip. Yes, OUM, FRAM, MRAM, and all the others face challenges when it comes to integration with standard (i.e., CMOS) silicon processes. In the meantime, stacked die is an option. In any case, suppliers claim that none of the issues are showstoppers and all are targeting future integration with standard logic functions. Doing so would highlight the advantages of the new memory technologies (e.g., flexibility, speed, endurance, and power consumption) while diluting the disadvantages (density and price per bit).

Flash memory and DRAM will be around a long time. But that doesn't mean the new technologies have to wait around. It's not a zero sum game, and there are plenty of existing and as yet unforeseen applications to support a variety of new technologies. I say bring 'em on.

Tom Cantrell has been working on chip, board, and systems design and marketing for several years. You may reach him by e-mail at tom.cantrell@circuitcellar.com.

REFERENCES

[1] "Elpida Memory Ships Samples of 2 Gigabit DDR2 SDRAM using 80 nm Process Technology," www.physorg.com/news9189.html.

[2] R. Neale, D. Nelson, G. Moore, "Non-Volatile and Reprogrammable, the Read-Mostly Memory is Here," *Electronics*, September 1970.

[3] J. L. Griffin et al, "Operating Systems Management of MEMS-based Storage Devices," Fourth USENIX OSDI Symposium, www.usenix.org/publications/library/proceedings/osdi2000/full_papers/griffin/griffin_html/index.html.

[4] M. Onken, Admin, MadSci Network, 2000, www.madsci.org/posts/archives/2000-10/971190308.Ch.r.html.

[5] F. Chu and T. Davenport, "The Endurance Performance of 0.5um FRAM Products," Ramtron International Corp., www.ramtron.com/lib/literature/techpapers/Endurance%20of%200.5u%20FRAM%20Products.pdf.

RESOURCES

Ovonic Unified Memory and Polymeric Ferroelectric RAM, Intel Corp., www.intel.com.

S. Lai, "Non-Volatile Memories: A Look Into the Future", Intel Corp., ftp://download.intel.com/technology/silicon/Stefan-SRI-Stanford-Berkeley-MIT-022604.pdf.

MEMS Ovonic Unified Memory Disk, Nanochip, Inc., www.nanochip.com.

Ovonic Unified Memory, Ovonyx, Inc., www.ovonyx.com.

Matrix 3D One-Time Programmable Memory, SanDisk Corp., www.sandisk.com.

SOURCES

Magnetoresistive RAM
Freescale Semiconductor, Inc.
www.freescale.com

Ferroelectric RAM
Ramtron International Corp.
www.ramtron.com

IDEA BOX

THE DIRECTORY OF PRODUCTS AND SERVICES

AD FORMAT: Advertisers must furnish digital submission sheet and digital files that meet the specifications on the digital submission sheet. **ALL TEXT AND OTHER ELEMENTS MUST FIT WITHIN A 2″ × 3″ FORMAT**. Call for current rate and deadline information. Send your disk and digital submission sheet to: IDEA BOX, Circuit Cellar, 4 Park Street, Vernon, CT 06066 or e-mail adcopy@circuitcellar.com. For more information call Shannon Barraclough at (860) 872-3064.

The Suppliers Directory at www.circuitcellar.com/suppliers_dir/
is your guide to a variety of engineering products and services.

PHYTEC

phyCORE® OEMable Single Board Computers

8051 CAN

C166/XC16x/ST10

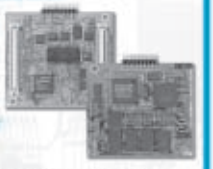
PowerPC

x86:

ColdFire:

XScale:

ARM7:



- sub-miniature SBC subassemblies (as small as 47x55 mm)
- low EMI, multi-layer design plugs like "big chip" into target circuitry
- 32 KB to 128 MB external SRAM and Flash (controller-dependent)
- CAN, Ethernet, RS-232/485; USB, IrDA; ADC, DAC (controller-dependent); free /CS and I/O
- available in w/demos and compilers
- insert-ready COTs subassemblies accompany you from evaluation through prototyping to end production... accelerating your design cycle and reducing time-to-market

(800) 278-9913 ■ www.phycore.com

USB

Add USB to your next project—it's easier than you might think!

- USB-FIFO up to 8 mbps
- USB-UART up to 3 mbps
- USB/Microcontroller boards pre-programmed with firmware
- 2.4GHz ZigBee™ & 802.15.4
- RFID Reader/Writer

Absolutely NO driver software development required!

www.dlpdesign.com

DLP Design

Start designing with Freescale's advanced S12X family!

Technological Arts

NEW!



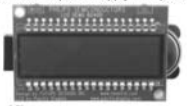
freescale Alliance Member

Toll-free: 1-877-963-8996
www.technologicalarts.com

USB-ICP & LCD-Demo Kit



USB-ICP Using a standard USB port, USB-ICP supports In-System Programming (both ISP and ICP mode) for Philips 80C51, LPC9xx, and LPC2xxx ARM7 microcontroller families. In-System Programming uses a two-wire serial interface to program and erase ISP enabled microcontroller devices without removing them from the system. This device is powered over the USB connection, so no external power supply is required! **$69.00**.



LCD-Demo Kit Small footprint "pocket" LCD Demo board with 8-character alphanumeric LCD. Uses a highly integrated Flash based 80C51 device. A single CR2025 3V coin cell battery powers the entire board and the unit is User Re-Programmable via the ICP connector. **$49.00.**

For XA Development Kits and I2C, MDIO and SPI tools, please visit our website.

FDI
www.teamfdi.com
VISA/MC/Amex

Future Designs, Inc.
2702 Triana Blvd
Huntsville, AL 35805
(256) 883-1240
Fax (256) 883-1241

AVOCET SYSTEMS, INC.

Hardware and Software Development Tools

683xx
PowerPC
68HC11
EZ80
Z80
8051
68HC12



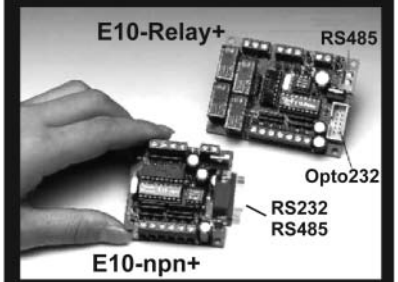
- Background Mode Debuggers
- In-Circuit Emulators
- C Compilers
- Assemblers

AVOCET

(207) 596-0080
sales@avocetsystems.com
www.avocetsystems.com

The $69 PLC

Work as Stand-Alone Ladder Logic PLC.
Or as Smart Remote I/Os of PC/ PLCs.
RS485 allows 256 units to be networked.



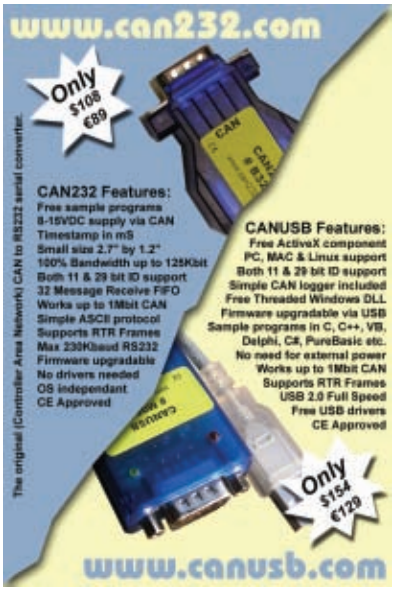
Incredibly Easy to Program!
Our software is used by many colleges for teaching PLCs!

Get Free Ladder Logic Simulator:
www.tri-plc.com/cci.htm

TRI Triangle Research

Tel: 1-877-874-7527 - PLC specialist since 1993

www.can232.com



www.canusb.com

ANSI C software development tools for:

IMAGEcraft™

NEW!: Global Optimizer

Version 7 Tools
Full-featured 45-day demos on our website

CypressM PSoC
AVR & tinyAVR
HC08,11,12,16
TI MSP430
ARM

hardware kits:

Elektronikladen - HC08 & HCS12 / 12 ref. design & starter kits
CANDIP - AVR/CAN on a chip!
Ethernut - Complete AVR TCP/IP solution

high powered development, garage powered prices!

www.imagecraft.com
info@imagecraft.com
706 Colorado Ave. Palo Alto, CA 94303
(650) 493-9326 • FAX (650) 493-9329

VISA / MC / AMEX
Check / Money Order

...say 'NO' to JURASSIC PRICES!

Get The Whole Picture

Camera Interface Application Kit for remote monitoring, security and event-capture via the internet.

Kit includes:
- RCM3365
- Serial Camera
- Servos
- Dynamic C®
- Dev. Tools

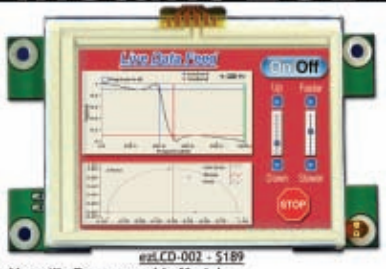


RABBIT Semiconductor

www.rabbitappkits.com

06231

Instant LCD



eaLCD-002 - $189

- Versatile Programmable Module
- Serial/I²C/USB/Parallel Interface
- AVR/BASIC Stamp/VB Compatible
- Onboard Flash Bitmap Memory
- Downloadable TTF Fonts
- 2.7" or 5.6" TOUCH Color TFT LCD
- 240x160 or 320x240 resolutions
- Transflective w/ LED Frontlight (2.7")
- Transmissive w/ 350 nit backlight (5.6")
- 512 colors or 65,535 colors

EARTHLCD.com We Make LCDs Work.™

JTAG & In-Circuit EMULATORS
IDEs, Compilers, Programmers



Phyton 718.259.3191 www.phyton.com

NEED AN INTERFACE TO YOUR KEYBOARD PORT?

INTERFACE:
- KEYPADS
- SWITCHES
- RS-232

TO YOUR PC'S KEYBOARD OR USB PORT

888-690-9080
HAGSTROMELECTRONICS.COM
11 Fiddlers Green, Lansing NY 14882

Solve complex signal acquisition problems...

- positioning & control
- environmental
- acceleration
- transients
- pressure
- vibration
- sonar
- GPS



- Linux Driver
- Guaranteed in stock
- Customization available
- 16-bit analog inputs and outputs
- Million sample FIFO eliminates interrupts
- Wide analog input and output ranges
- -40°C to +85°C Standard

www.stx104.com
Apex Embedded Systems
sales@stx104.com • 608-256-0767 x24

I²C™/SMBus



- Bus Monitors
- Protocol Analyzers
- Host Adapters
- Multiplexers
- Battery Applications
- Software Tools

MCC Micro Computer Control

I²C is a trademark of Philips Corporation

www.mcc-us.com

NEW! CAN-4-USB FX
USB to CAN Interface
USB 2.0 Hi-Speed 480Mbps!
Other companies may claim USB 2.0 but if it isn't Hi-Speed it is only 12Mbps.
2006 marks our 10th year of selling CAN interfaces in over 30 countries!
$249 USD.

Zanthic Technologies Inc.
403-526-8318
www.zanthic.com

IC DEVICE AND PACKAGE CONVERTERS

* Upgrade Your PCB
* Fix Design Problem
* Replace Obsolete Part
Our experienced engineers can Quickly solve your problem. We provide solutions for all IC package types including BGA, QFP, PLCC, SOIC, DIP, PGA.
Ironwood Electronics
1-800-404-0204
www.ironwoodelectronics.com

GoFast™ MATH PACKAGE
- Improve floating point speed without increasing hardware cost
- Replaces compiler math functions
- Intrinsic functions 2-3X faster
- Complex functions 5-10X faster
- In assembly language for speed
- Many processors supported
- Conforms to IEEE 754
- Full source code, no royalties
www.smxrtos.com
Micro Digital Inc
RTOS Innovators
800.366.2491 sales@smxrtos.com

Affordable Powerful & Portable
USB Lab Instruments
Oscilloscope $199US
Dual Channel, 20 MS/s
Hardware Preamp w/ 7 Ranges
Pre & Post Trigger Capture
Automatic Measurements
8-bit Vertical Resolution
Variable Trigger Level
Auto, Normal and Single-Shot Trigger

Waveform Generator $139US
Arbitrary DDS Waveform Generation
Four Built-In Waveforms:
Sine, Square, Triangle, Sawtooth
One User Programmable Waveform
0.1Hz to 100kHz
8-Bit Resolution
Amplitude and Offset Adjustments

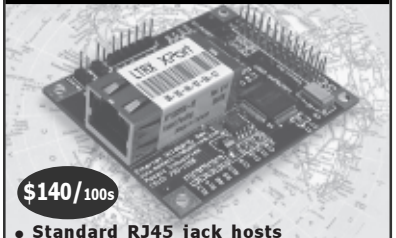
Open Source Software ■ Windows & Linux
Bonus: Network analyzer software
See our website for details
www.syscompdesign.com
647-839-0325 Toronto, Canada

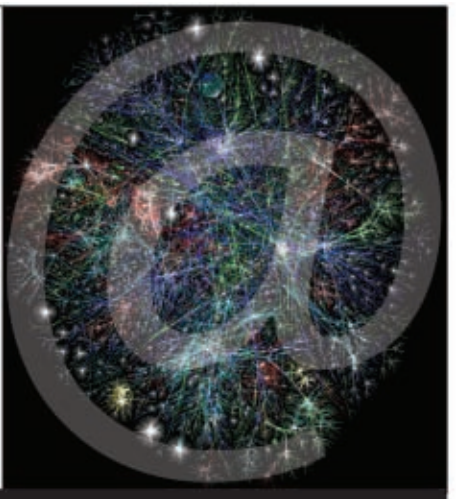
Full Speed CAN USB Adapter
✓Simple configuration & use
✓10% Discount for first orders Code: GRIDCC20
$235.00 Qty 1


+1 630-245-1445
Naperville, Illinois USA
www.c-a-n.com
gridconnect

Controllers
NEW ET-AVR Stamp Module


- Includes ATMega64 Microcontroller
- 64kbytes Flash Program Memory
- Up to 53 I/O Points
- 8 Channel 10-bit A/D
- Ideal As plug-in Controller
- Direct Program Download
Also Available: ET-AVR Stamp Development Board with P/B's, LED's, LCD Connector etc
www.futurlec.com

EtherSmart Wildcard™
Network -Enables Your Product

$140/100s
- Standard RJ45 jack hosts 10/100Mbit Ethernet
- HTTP, SMTP, TCP, DHCP, ICMP, and ARP Protocols
- Email program-controlled messages to a specified LAN IP address
- Establish a TCP/IP connection to exchange binary or ASCII data
- Serve software-controlled dynamic content to your web browser
Mosaic Industries Inc.
tel: 510-790-1255 fax: 510-790-0925
www.mosaic-industries.com

CIRCUIT CELLAR
Designer's Notification Network
Circuit Cellar design contest entrants have received thousands of valuable development tools and product samples. Because of their contest participation, these engineers receive advance e-mail notice from Circuit Cellar as soon as new samples become available. Now you too can benefit from this early notification.
Welcome to the Designer's Notification Network. Print subscribers are invited to join the Network for advance notice about our new sample distribution programs.

Find out more at www.circuitcellar.com/network.

MYLYDIA, INC.

Layout Gerber,
Prototype Making

QUICK TURN

PCB & Turnkey
@
The Best Prices

Tel: 800-695-9342

Sales@mylydia.com

WWW.MYLYDIA.COM

CHINA PCB SUPPLIER

DIRECT SALE FROM CHINA
PROTOTYPE TO PRODUCTION
instant online quote
shopping cart ordering system
China competitive prices
free Electrically test



WWW.PCBCART.COM

SmartHomeUSA



SAVE 10%
ENTER COUPON CODE CELLAR DURING CHECKOUT

www.SmartHomeUSA.com
DIY Home Automation, Security & Entertainment Products

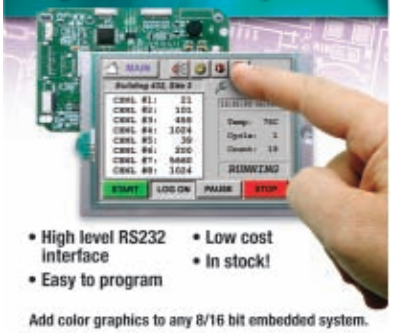
CUSTOM MEMBRANE KEYBOARDS / SWITCHES



- 1 TO 2 WEEKS TURNAROUND
- VERY COMPETITIVE PRICING
- Ex.: (5) 4-switch keyboards for $395.00
- PCB backed switches
- Custom metal backplates/assemblies
- Electronic assemblies/graphic overlays
- Electronic file transfer capabilities

Picofab Inc.
4780B Blvd. Henri-Bourassa
Charlesbourg, Quebec, Canada G1H 3A7
Tel: (418) 622-5298 • Fax: (418) 622-9996
Email: sales@picofab.net

Add a color touch interface to your embedded product!



- High level RS232 interface
- Low cost
- In stock!
- Easy to program

Add color graphics to any 8/16 bit embedded system.
Easy, fast and flexible. Up and running in hours!

REACH TECHNOLOGY INC.
www.reachtech.com • 510-770-1417
842 Boggs Avenue • Fremont, CA 94539

PCBFABEXPRESS
High Quality PCBs @ Low Impact Prices

Get 5 PCBs for $13 each in 5 days

2 layer, 20 sq. in., FR-4, 0.062" thick
FREE Tooling, FREE Soldermask, FREE Silkscreen

Visit website to see all offers on 2 to 8 layer PCBs

ORDER ONLINE
408-522-1500
www.PCBFabExpress.com

Also see our fantastic PCB Assembly prices online

SpectraPLUS 5.0
Audio Spectrum Analysis

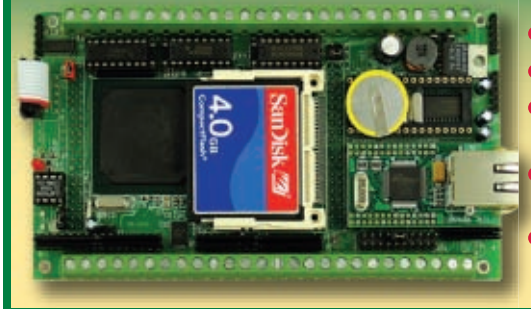
Features
Sound Card based I/O
FFT sizes to 1048576pts, 1/96 Octave
Up to 24 bit, 200kHz sampling rates
3-D Surface and Spectrogram
Digital Filtering, Signal Generation
THD, IMD, SNR, Transfer Functions
DDE, Macros, Data Logging,
Vibration Analysis, Acoustic Tools

FREE 30 day trial!
www.spectraplus.com

PHS
Pioneer Hill Software
360 697-3472 voice
pioneer@telebyte.com

5D™ 586-based Industrial Controller
High-performance Ethernet/TCP, 24-bit ADC, DAC, HV I/O, and CF Interface.

OEM $199



- 151x82 mm. DIN rail mounting
- AMD SC520, program in C/C++
- 4 RS232/485, ADC, DAC, Solenoid Drivers, OPTO
- CompactFlash and FAT16 file system support
- Hardware TCP/IP stack for 100M Base-T Ethernet

50+ Low Cost Controllers with ADC, DAC, solenoid drivers, relays, PC-104,CF, LCD, DSP motion control, 10 UARTs, 300 I/Os. Custom board design. Save time and money.

TERN INC. 1724 Picasso Ave., Suite A, Davis, CA 95618 USA
Tel: 530-758-0180 • Fax: 530-758-0181 www.tern.com • sales@tern.com

ValueCAN

The High Value Tool
For Controller Area Network

- USB to CAN
- Simple software analyzer included
- DLL with examples for custom apps
- PC isolated from CAN
- 100% bandwidth at 500Kb

Only $295

ICS RADDAQ

www.intrepidcs.com/vcan

8/16-bit EEPROM | Ser. EEPROM | Flash EPROM | GAL / PALCE | Most MCU's | Low Voltage to 1.3V. | DIL dev. w/o Adapter.

Conitec's last generation Galep-4 employs ASIC universal pin technology for each pin of 40 pin ZIF-socket. **9000+device library** / lifetime free updates. Programs 8/16 bit EPROM'S, EEPROM's, 0-Pwr RAM, FLASH, Serial EEPROM's, GAL, PALCE, microcontrollers such as 87/89xxx, PIC, AVR, ST62, etc. Low voltage devices down to 1.3V. No adapter required for DIL devices. 8 Hrs. operation on battery (AC charger included). Runs **WIN 98,NT,ME, 2000, XP** with Hex/Fuse Editor. Remote control from other apps, (e.g. VisualBasic). Substitutes high priced universal programmers e.g. ALL-11 (HILO) or LAB-TOOL-48 (ADVANTECH) **Provides virtually matching performance at 1/3 to 1/5 the price.** Info, orders, softwr : **619-462-0515**

A PALM-SIZED PROGRAMMER HANDLES OVER 9000 DEVICES ?

ISO 9001:2000 CERTIFIED PROFESSIONAL TOOLS

9000 DEVICES



SMALL PACKAGE. BIG FEATURES.

SALES, SUPPORT / EMAIL: CONTACT@CONITEC.NET / WWW.CONITEC.NET TEL: 619-462-0515. FAX: 619-462-0519



Tiny Linux controller

16MB fast SDRAM	10/100 Ethernet
64MHz Coldfire MCU	3 serial ports (RS232)
4.5 MB flash memory	1 CAN port
SD card socket (to 2GB)	LCD/KPD port
Eclipse/CDT dev. env.	Free serial debugger

55 I/O pins & capabilities to burn...

www.steroidmicros.com

PROTOTYPE CIRCUIT BOARDS

$85 US for two 5" x 6" 2-layer boards

Shipped NEXT BUSINESS DAY if data is received by 1:00 pm EASTERN

www.apcircuits.com

AP Circuits

(403) 250-3406

VISA MasterCard staff@apcircuits.com



CIRCUIT CELLAR®

back issues available as

Searchable Archives on CD-ROM

NOW SHIPPING:

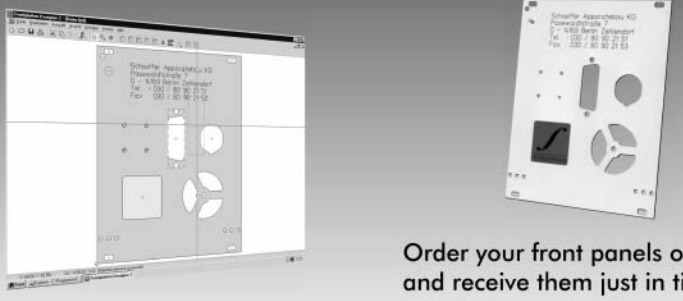
CD-ROM #10 2005 Issues 174-185
CD-ROM #9 2004 Issues 162-173
CD-ROM #8 2003 Issues 150-161

Order Online:
www.circuitcellar.com
or call 860.875.2199

Front Panels?

Unrivaled in price and quality for small orders

Download the free ›Front Panel Designer‹ to design your front panels in minutes



Order your front panels online and receive them just in time

www.frontpanelexpress.com

Serial • Ethernet
Web Server

Simple, DTE, DCE
No SW Changes



+1 630-245-1445
Naperville, Illinois USA
www.gridconnect.com
gridconnect

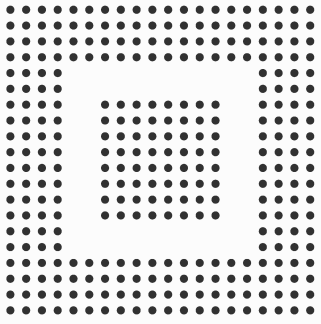
FAT 12/16/32
FILE SYSTEM

- DOS/Windows Compatible
- USB Flash Disk
- USB Floppy & Hard Disk
- SD/MMC
- CompactFlash, ATA/IDE
- DiskOnChip
- NAND & NOR Flash
- 10KB RAM / 25KB ROM typical
- Low Cost, No Royalty
- Full Source Code

www.smxrtos.com
Micro Digital Inc
RTOS Innovators
800.366.2491 sales@smxrtos.com



Weather
Instruments
for PCs
AAG electrónica
www.aagelectronica.com



Start at $20
*BGA Mounting
*BGA Repairing
*Ball Planting
WWW. EzPCB. COM

Flashlite
186



- 186 processor = 33 MHz
- DOS w/ Flash File system
- 44 Digital I/O lines w/ CPLD
- Console / Debug Serial Port
- 7-34V DC or 5V DC power • 2 Serial Ports
- Accepts 8MB DiskOnChip • 2 16-bit Timers
- 512K DRAM & 512K Flash • Watchdog Timer
- Expansion options with Peripheral Boards

$69 QTY 1

$99 Development System

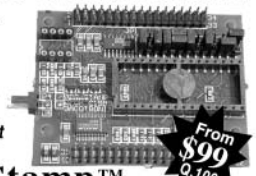
Development kit includes:
- Flashlite 186 controller
- Borland C/C++ ver 4.52
- FREE Email Tech Support
- Serial Driver library
- AC Adapter and cable

Call 530-297-6073 Email sales@jkmicro.com
On the web at www.jkmicro.com
JK microsystems

NOW YOU CAN HAVE IT ALL!

- Easy to Use
- Low Cost
- Low Power
- Tiny Size
- High Speed
- DOS Environment

with the
DOS Stamp™



From $99 Q.100

Easy Software Development: Use your C/C++ or Basic compiler to produce DOS EXE. Download EXE to flash disk via serial port. EXE runs on power up.

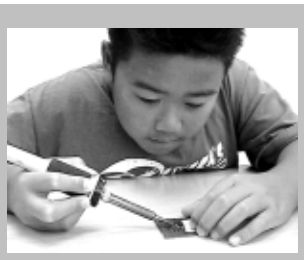
Standard Features: BIOS & DOS-ROM, 128K flash disk, 512K SRAM, 40 MHz CPU, 16 digital I/O (opto rack interface), 2 RS-232, 2 timer/counters, simple bus interface, real-time clock

Options: 8-ch 12-bit ADC, flash disks up to 288 MB, 1 RS-485, TCP/IP

Tiny Size, Low Power: 2"x2.6", 5V @ 200 mA at full speed, 40 mA at lowest speed.

Visit http://www.bagotronix.com for info, prices, and FAQs

BAGOTRONIX
2900-1 Crescent Drive
Tallahassee, FL 32301
850-942-7905 phone & fax



ANYONE
Can Now Easily
Hand Solder Surface-
Mount Components!
Even A 10
Year Old!
www.schmartboard.com

PRINTED CIRCUIT BOARDS
QUALITY PRODUCT
FAST DELIVERY
COMPETITIVE PRICING

PROTOTYPE through PRODUCTION

- Aluminum Backed PCB
- Single & Double sided
- SMOBC/RoHS
- LPI mask
- Through hole or SMT
- Nickel & Gold Plating
- Routing or scoring
- Electrical Testing
- Artwork or CAD data
- Fast Quotes
- Flex Circuits

SPECIAL OFFER:
10 pcs (3days) 1 or 2 layers $249
10 pcs (5days) 4 layers $695
(up to 30sq. in.ea.) includes tooling, artwork, L.P.I. mask & legend

PULSAR, INC
9901 W. Pacific Ave. Franklin Park, IL 60131 • Phone 847.233.0012
Fax 847.233.0013 • www.pulsar-inc.com • sales@pulsar-inc.com

Full Speed
It writes your USB Code!

NEW! HIDmaker FS for Full Speed FLASH PIC18F4550

Creates complete PC and Peripheral programs that talk to each other over USB. Ready to compile and run!

- Large data Reports
- 64,000 bytes/sec per Interface
- Easily creates devices with multiple Interfaces, even multiple Identities!
- Automatically does MULTITASKING
- Makes standard or special USB HID devices

NEW! "Developers Guide for USB HID Peripherals" shows you how to make devices for special requirements.

Both PC and Peripheral programs understand your data items (even odd sized ones), and give you convenient variables to handle them.

PIC18F Compilers: PICBASIC Pro, MPASM, C18, Hi-Tech C.

PIC16C Compilers: PICBASIC Pro, MPASM, Hi-Tech C, CCS C.

PC Compilers: Delphi, C++ Builder, Visual Basic 6.

HIDmaker FS Combo: Only $599.95

DOWNLOAD the HIDmaker FS Test Drive today!

Trace SYSTEMS, Inc.

www.TraceSystemsInc.com

301-262-0300

ALL ELECTRONICS CORPORATION

Electronic and Electro-mechanical Devices, Parts and Supplies.
Wall Transformers, Alarms, Fuses, Relays, Opto Electronics, Knobs, Video Accessories, Sirens, Solder Accessories, Motors, Heat Sinks, Terminal Strips, L.E.D.S., Displays, Fans, Solar Cells, Buzzers, Batteries, Magnets, Cameras, Panel Meters, Switches, Speakers, Peltier Devices, and much more....

www.allelectronics.com

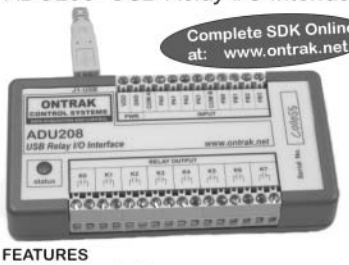
Free 96 page catalog

1-800-826-5432

USB Data Acquisition

ADU208 -USB Relay I/O Interface

Complete SDK Online at: www.ontrak.net

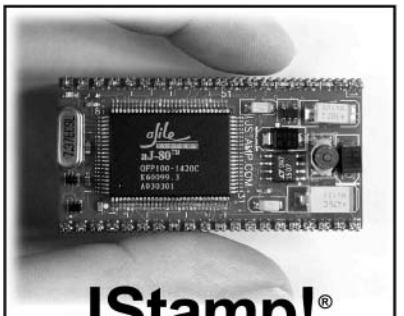


FEATURES
-8, 5-AMP relay outputs
-8, ISOLATED digital inputs
- Port Powered, Aux 5VDC output $189.00 QTY 1

Other Models;
ADU200- 4 Channel Version with RS232 $139.00
ADR218- Solid-State Version 8 Channel $225.00
ADU100- 3 CH, 16-Bit ISOLATED Analog Inputs, PGA, 4 digital I/O, RS232 and 5 AMP Relay Output $199.00

ONTRAK CONTROL SYSTEMS INC.
PH: (705) 671-2652 Fax: (705) 671-6127

www.ontrak.net



JStamp!®

FAST 32-bit native-execution Java
Executes 3 million Java byte codes/sec
compact 1 x 2 inch DIP40 package
Real J2ME CLDC Java (threads, floats)
2048 KB Flash & 512 KB SRAM

www.jstamp.com

SYSTRONIX
Salt Lake City, Utah, USA

Java is a TM of Sun Microsystems, Inc. JStamp is a TM of Systronix, Inc

---

INTRODUCING...

A new concept in Electronics Enclosures

NO drilling, NO punching, NO machining, NO customizing at all, EVER!



CircuitShell
www.CircuitShell.com

Check out our 8051 microcontroller evaluation board

Order online at: www.melabs.com

microEngineering Labs, Inc.

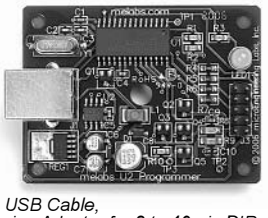
Development Tools for PIC® Microcontrollers

Phone: (719) 520-5323
Fax: (719) 520-1867
Box 60039
Colorado Springs, CO 80960

USB Programmer for PIC® MCUs — $119.95 (with accessories)

RoHS Compliant

Programs PIC MCUs including low-voltage (3.3V) devices



Includes:
Programmer, USB Cable,
ZIF Programming Adapter for 8 to 40-pin DIP,
Software for Windows 98/Me/NT/2K/XP



EPIC™

Parallel Port Programmer starting at $59.95

Serial Port Programmer starting at $79.95

LAB-X Experimenter Boards



Pre-Assembled Boards
Available for 8, 14, 18, 28, and 40-pin PIC® MCUs
2-line, 20-char LCD Module
9-pin Serial Port
Sample Programs
Full Schematic Diagram

Pricing from $79.95 to $349.95

PICPROTO™ Prototyping Boards



Double-Sided with Plate-Thru Holes
Circuitry for Power Supply and Clock
Large Prototype Area
Boards Available for Most PIC® MCUs
Documentation and Schematic

Pricing from $8.95 to $19.95

BASIC Compilers for PICmicro®



Easy-To-Use BASIC Commands
Windows 9x/Me/2K/XP Interface

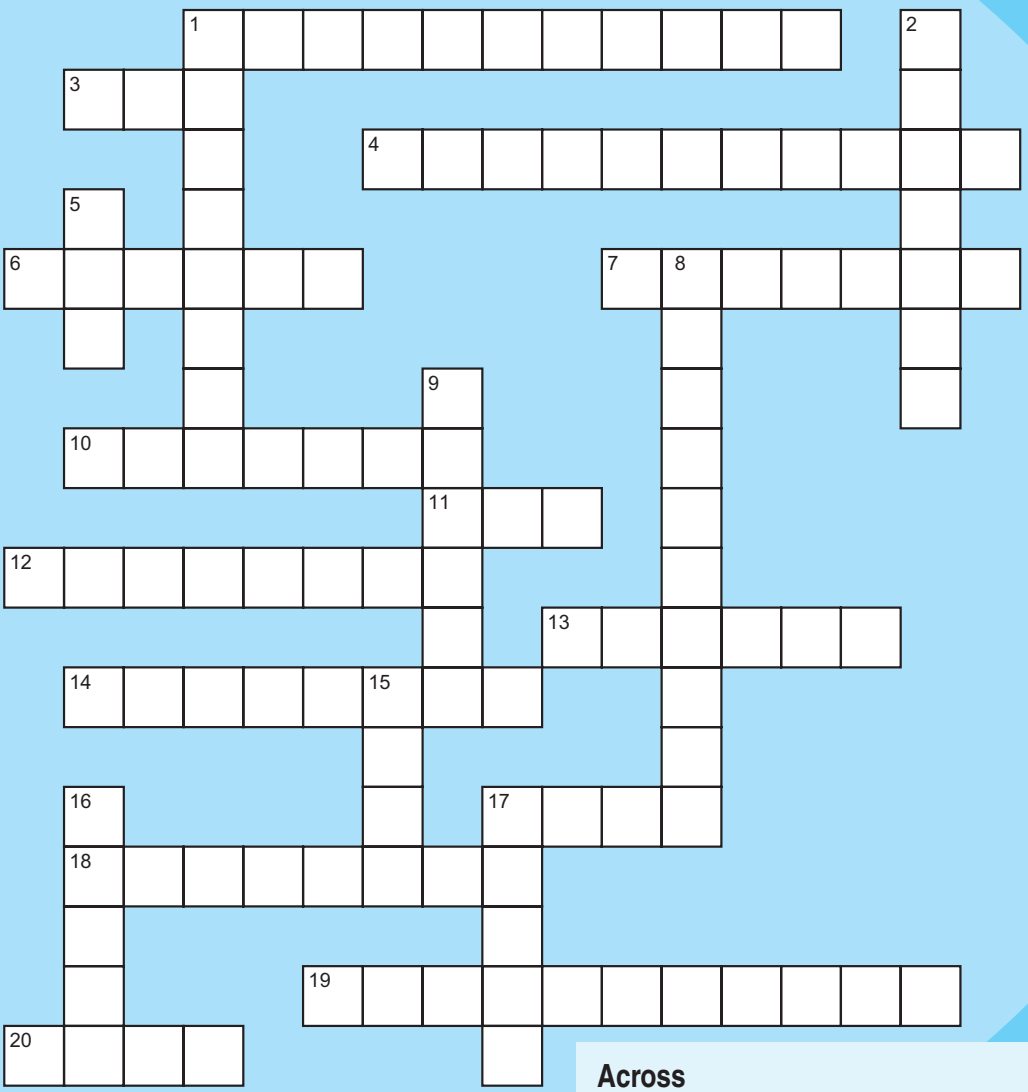
PICBASIC™ Compiler $99.95
BASIC Stamp 1 Compatible
Supports most 14-bit Core PICs
Built-In Serial Comm Commands

PICBASIC PRO™ Compiler $249.95
Supports All PICmicro® MCU Families
Direct Access to Internal Registers
Supports In-Line Assembly Language
Interrupts in PICBASIC and Assembly
Built-In USB, I2C, RS-232 and More
Source Level Debugging

See our full range of products, including books, accessories, and components at:

www.melabs.com

CROSSWORD



Down

1. The *H* in the following equation: H = E + PV. Hint: heat content
2. English electrical engineer (1849–1945) credited with designing the thermionic valve. Hint: 007
5. Encryption standard developed in 1976.
8. A type of robot designed to search for the sources of certain sounds.
9. A condition of motionlessness.
15. The name of the human-like creatures in H.G. Wells's novel *The Time Machine* (1895).
16. }:>
17. 0:>

The answers are available at www.circuitcellar.com/crossword.

Across

1. The state of balance in which opposing forces cancel each other out.
3. A thin metal prong found on MCUs and connectors.
4. An instrument for measuring absorbed and discharged heat.
6. The U.S. state in which Area 51 is located.
7. Latin: *causticus* (combustible or burning)
10. NASA robot used for deep ice exploration.
11. Away from keyboard
12. ANSI code: 0133
13. A UNIX or DOS directory located above a "child."
14. To make digital information more compact and thereby take up less space.
17. German scientist (1840–1905) who designed the first refractometer. You might have used the prism that bears his name.
18. A layer of silver bromide and gelatin that coats photographic film.
19. A room or building that's used for observing astronomical occurrences.
20. The color commonly associated with IBM. Hint: "Big *xxxx*"

INDEX OF ADVERTISERS

The Index of Advertisers with links to their web sites is located at www.circuitcellar.com under the current issue.

Page		Page		Page		Page	
91	AAG Electronica, LLC	39	Echelon Corporation	47	Lantronix	89	Pioneer Hill Software
90	AP Circuits	55	Efficient Computer Systems	87	Lawicel HB	91	Pulsar, Inc.
74	AVR 2006 Design Contest	8	Elprotronic, Inc.	41	Lemos International	23, 49	Rabbit Semiconductor
11	Accurate Technologies, Inc.	26	ExpressPCB	64	Linx Technologies	87	Rabbit Semiconductor
92	All Electronics Corp.	91	ezPCB	8	Logic Supply, Inc.	89	Reach Technology, Inc.
87	Apex Embedded Systems	86	FDI-Future Designs, Inc.	2	Luminary Micro	50	RoboDevelopment Conf. & Expo
24	Arcom Control Systems	18, 19	Freescale Semiconductor, Inc.	87	MCC	91	Schmartboard
7	Atmel	90	Front Panel Express	95	MVS	40	Sealevel Systems
63	Avnet Electronic Marketing	88	Futurlec	15	Matrix Orbital	9	SEGGER Microcontroller Sys. LLC
86	Avocet, Inc.	89	General Circuits, Inc.	11	Maxstream	22	Sierra Proto Express
91	Bagotronix, Inc.	88, 91	Grid Connect	88, 91	Micro Digital, Inc.	89	Smart Home Systems
71	Bitscope Designs	87	Hagstrom Electronics	29	Microchip	27	Spark Fun Electronics
65	CWAV	73	HI-TECH Software LLC	92	microEngineering Labs, Inc.	88	Syscomp Electronic Design Ltd.
56	CadSoft Computer, Inc.	87	IMAGEcraft	57	mikroElektronika	92	Systronix
92	Circuit Shell	10	Infineon	88	Mosaic Industries, Inc.	C3	Tech Tools
3	Comfile Technology, Inc.	90	Intec Automation, Inc.	5	Mouser Electronics	80, 81	Technologic Systems
90	Conitec	90	Intrepid Control Systems	89	Mylydia, Inc.	86	Technological Arts
90	Custom Computer Services, Inc.	31	Intronix Test Instruments, Inc.	C2	NetBurner	89	Tern, Inc.
61	Custom Solutions, Inc.	88	Ironwood Electronics	69	Nurve Networks LLC	49	Tianma Microelectronics
1	Cypress MicroSystems	64, 91	JK microsystems, Inc.	92	Ontrak Control Systems	48	Tibbo Technology, Inc.
86	DLP Design	82	Jameco	89	PCB Fab Express	92	Trace Systems, Inc.
60	Decade Engineering	60	Jeffrey Kerr LLC	C4	Parallax, Inc.	33	Tri-M Systems, Inc.
61	EMAC, Inc.	22	Keil Software	86	Phytec America LLC	86	Triangle Research Int'l, Inc.
87	Earth Computer Technologies	69	LabJack Corp.	87	Phyton, Inc.	88	Zanthic Technologies, Inc.
25	EasySync Ltd.	69	Lakeview Research	89	Picofab, Inc.		

Preview of December Issue 197
Theme: Analog Techniques

ARM Scope: Build an Analog Oscilloscope Companion

QuickComs: An M16C/62P-Based RS-232 Analyzer

Digital Instrument Panel: A Simple Panel for Diesel Engines

Visual Basic 2005 and the Serial Port

Multitiered Security System

Third Overtone Crystal Clock Oscillator

ABOVE THE GROUND PLANE **Filter Figures**

APPLIED PCs **Ethernet on a Chip**

FROM THE BENCH **Portable Lightning Monitor**

SILICON UPDATE **Ferro 51**

ATTENTION ADVERTISERS

January Issue 198 Deadlines

Space Close: Nov. 13
Material Close: Nov. 21

Theme: Embedded Applications

BONUS DISTRIBUTION: DesignCon West

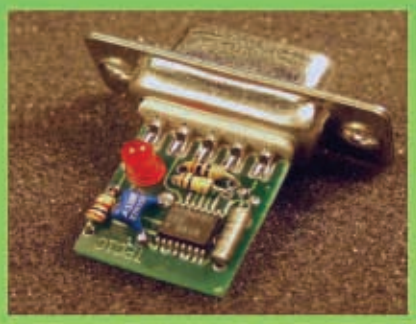
Call Shannon Barraclough now to reserve your space!
860.872.3064
e-mail: shannon@circuitcellar.com

$25 EVAL (QTY 1)
$7.10 OEM (1K)

TINY PC!

1/2" FORM FACTOR!
RUN X86 ASSEMBLY, TURBO C OR QBASIC.
EASY TO USE!

TPC1 is a functional computer complete with CPU, memory and I/O. Not a DIP module, core or "engine". No special cables or development kits to buy. Download programs with XP Hyperterm. Micro-power, run from serial line or battery.



- Built-in FLASH, RAM, EE
- Up to 6 serial or parallel I/O
- On-board diagnostic LED
- PC compatible BIOS routines
- Hardware Real-Time-Clock
- Up to 4 channels 12bit ADC
- Ready to go DB9 connector

TPC2 (more mem/io) $50/$14

TRUE FRAME GRABBER

-Color & BW up to 640x480
-Serial/parallel:pic,avr,z80,pc
-Full speed: to 30 frames/sec
-Simultaneous composite out
-Use w/digital CMOS camera
-C, BASIC, Assembler source

oem(1k) **$27**
eval kit(1) **$95**



Add vision at low cost! Perfect for inspection, security,robotics. Full frame buffer unlike sx, pci,usb type. Industry std PC104 form factor.

CREDIT CARD COMPUTER 2



New PLUG-N-GO, no cables/power supply to buy! Lo-power RISC cpu 10x faster than PIC, Z80, 8051 4meg NV mem,ser,par,RTC,4ch 12bit ADC,ISA bus Built-in BASIC, Assembly, C compiler included Friendly instruction set, unlike PIC or 8051
CCC2 eval(1)$75/oem(1k)$21 CCC1 $50/$14

VOLKSCOMPUTER

A CONTROLLER FOR EVERYONE!

PC compatible complete with keyboard & display. Debug programs on the desktop then download to VolksComputer. Super bright LCD. Tactile-click keypad accesses entire alphanumeric char set. 10 bit ADC. Serial (3f8-2e8) and parallel (378h) ports.
VC1: eval $95 oem $27 VC2: $150/$42



SERIAL MINI-TERMINAL

RS232 terminal for Stamp, PC, Z80, AVR etc.
-super low-current, powers from serial line
-LED backlit LCD, visible in all conditions
-115.2kbps, DB9 conn, simple commands
-specify 20 customizable or 16 tactile keys
eval(1) $75,oem(1k) $21.30,w/BASIC cpu $27



640x480 VGA LCD $27

Controller for most single/dual scan LCDs
Works with lo-res (160x120, 320x240,etc.)
Use with PC or SBC, standard VGA BIOS
Source code demo shows VGA initialization
Adaptable for other CPUs (i.e. Z80, HC11)
oem(1k) $27 evalkit(1) $95 w/10"LCD $195

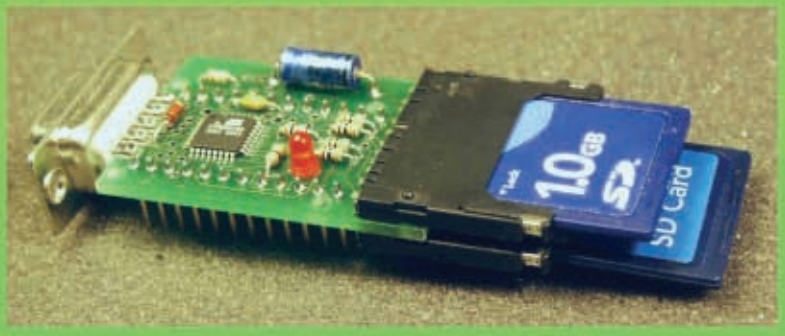


RFID MODULE $7



New lower cost unit has longer range and reads more tag types than competitive units. On-board power FET drives solenoid directly with card qualify functions. Serial out.
RFID2 eval (qty 1) $25, oem (1k) $7.10

RS232 TO SD FLASH



HALF THE SIZE, HALF THE COST, TWICE THE VALUE!

Read/Write PC compatible media with Stamp, Z80, AVR, PIC micros. Up to 128 files at once 16 gigabyte max size.

SD1 oem (1k) price $11.35,
SD2 $75/$21

$39.95 single piece price

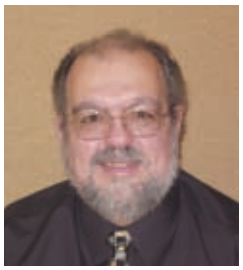
WWW.STAR.NET/PEOPLE/~MVS
Alternate: WWW.GEOCITIES.COM/MEGAMVS

visit website or call for more info (508) 792 9507

.MVS

5yr Limited Warranty
Free Shipping
Mon-Fri 10-6 EST

SERVING THE EMBEDDED COMMUNITY SINCE 1979!



PRIORITY INTERRUPT

by Steve Ciarcia, Founder and Editorial Director

In Search of the Real Thing

I had an interesting experience recently that I thought I should tell you about because it deals with a specific product and the ordeal I had finding it. Let me explain.

About two months ago, I installed a HomeVision-Pro home control system (www.csi3.com/homevis2.htm). It has a rather nice PC-based GUI for programming, but all report generation and event logging is done via an integral NTSC video display with an IR remote and a couple auxiliary serial ports. There was a web-based GUI and Ethernet interface for network communication (by a third-party supplier), so I felt my goal of remotely viewing and controlling the HCS could be achieved. Unfortunately, my experience thus far is that it would be easier to build a Star Trek matter transporter than get this "glued-on" web interface working. (In truth, my difficulties may be my ISP-supplied, non-user-programmable router and not them.) So, back to the drawing board to create a few "glue-ons" of my own.

Ultimately, remote real-time control was actually easier to do than I thought it would be. Because I have an extensive web-based video-monitoring system already installed, I simply attached the HV-Pro video and IR remote inputs to a commercial Sling Media Slingbox. It's not rocket science, but it works. The Slingbox thinks it's talking to an IR-controlled DVD player, but who cares? ;-)

Viewing the event log seemed like it should be even less complicated. I would simply dump the HCS event log out one of the serial ports and send it to a "dumb terminal" with an NTSC video display. Then I'd connect this video output to one channel on an Aruca Electronics video web server (www.arucaelectronics.com/shop/ipvideoserver.php) and look at it like any other video camera. Simple, huh?

After spending a couple hours on Google looking for things like "serial terminal," "RS-232 serial terminal," and "serial display," I concluded that I was in trouble. I found some small expensive hand-held test devices, lots of LCDs, and software emulations, but the conclusion was that the terminal world was now either Linux-based hardware or PC HyperTerminal variants. Forget any traditional hardware like a "dumb terminal" that used NTSC. Like the RS-232 serial port connector on my laptop, it had apparently gone the way of the Pentium II.

I was really frustrated at this point. Someone had to make a serial terminal with an NTSC video output, so it was back to the Internet where I spent an entire evening on Google. I think it was under "NTSC serial terminal" on page 36 of the search results that I finally came across Decade Engineering (www.decadenet.com). The irony is that Decade Engineering is a *Circuit Cellar* advertiser!!!

Well, I'm happy to say that the BOB-4 module is the best thing since sliced bread. I appreciate the product and it works great. But, there could be a bigger problem here. Decade advertised BOB-4 under my nose in my own magazine, and I didn't recognize what they were selling. Because I think of myself as a *Circuit Cellar* reader, I wonder if there are a bunch of other readers who are equally in the dark.

Basically, it's semantics. A butcher or grocery store can put up a big sign saying "Sliced Animal Products" or one that says "Beef Steaks." Obviously, there is a very large audience that understands that sliced animal products really means steaks. Or, momentum and brand recognition in a particular audience is such that it wouldn't matter what the sign said. People know to go to this butcher or grocer.

Many readers buy our magazine specifically for the ninth-page ads because it is like a catalog. Like most readers, I scan the ads and good ones catch my eye. The problem is whether the advertising language is so specific to one audience that it isn't immediately understood in a different audience. I specifically remember the Decade ad being about some "Basic Overlay Board." I'm embarrassed to say that I was so intent on solving my engineering problem that I perhaps focused too much on my own search nomenclature. I needed a serial terminal with NTSC.

In retrospect, I think that "Basic Overlay Board" led me in the wrong direction. Yes, I know what overlaying one NTSC video signal onto another is about, but an ad titled "BOB-4" didn't ring enough bells where I immediately understood that "BOB" really should be translated "serial ASCII video terminal." Yeah, I know. If I had studied the ad and put "video display generator" together with "SPI or RS-232 interface," I might have had a better shot. Too bad all of us don't have the time to study every ad in detail, but I didn't find it because I was on a search mission using *my* language and *my* nomenclature.

This all makes me wonder if we have any other advertisers who need to think about their advertising language. Our PC-obsessed world has left behind entire generations of industrial products in a twilight zone where PC simulations are the product of the day and it could be 35 search pages before anyone gets to your listing. These days, it may actually take special emphasis in an advertising message if you want people to know that you are still making the real thing. It sure would have made my life easier.

steve.ciarcia@circuitcellar.com



DigiView

100 MHz, 18 Channel, Portable Logic Analyzer

Auto Hardware Compression Captures
from 128K to 30 Billion Samples
@10ns resolution
only $499.00



Now with Serial Signal Analysis

Download and try the Software!

Cables & Micro-Clips Included

Professional Capture & Analysis Software Included

- Automatic Real-time Hardware Compression can save up to 5 minutes of data @ 10ns.
- Display I2C, Synchronous (SPI), Asynchronous (RS-232), State, Boolean, Bus and Analog Data.
- Edge and Pattern Triggers.
- Pattern Searches with Match, Duration & Animation.
- Specialized Sequential Searches for Serial and State Mode Signals.
- Drag & Snap Markers.
- Dual Waveform View with signal edge snapping.
- Multi-Signal Data Tables with Link Groups.
- Specialized Exports and List Views for Serial Signals.
- Print or Save Images with custom comments.
- USB Powered for Portability.
- USB 1.1 & 2.0 compatible.



PICmicro® MCU Programmer

Multi-Function In-Circuit & Gang Operation

only $199.00

- Supports 10, 12, 14, 16 & 18F Series PICmicro MCUs.
- Edit Code, EE Data, IDs and Configuration Fuses.
- Auto or Manual Serialization in Code or EE Data.
- Firmware auto synchronizes to Software version.
- Control Files protect Option Settings.
- Accepts Command-Line Parameters.
- Tests and Monitors Voltages.
- Control Signal for Self-Powered Circuits.
- Single and 4 Gang Adapters available.
- Includes Cables, Software & U.S. Power Supply.

EconoROM III™



EPROM & FLASH Emulation at its best!

from $179.00

FlexROM III™







QuickLoader™

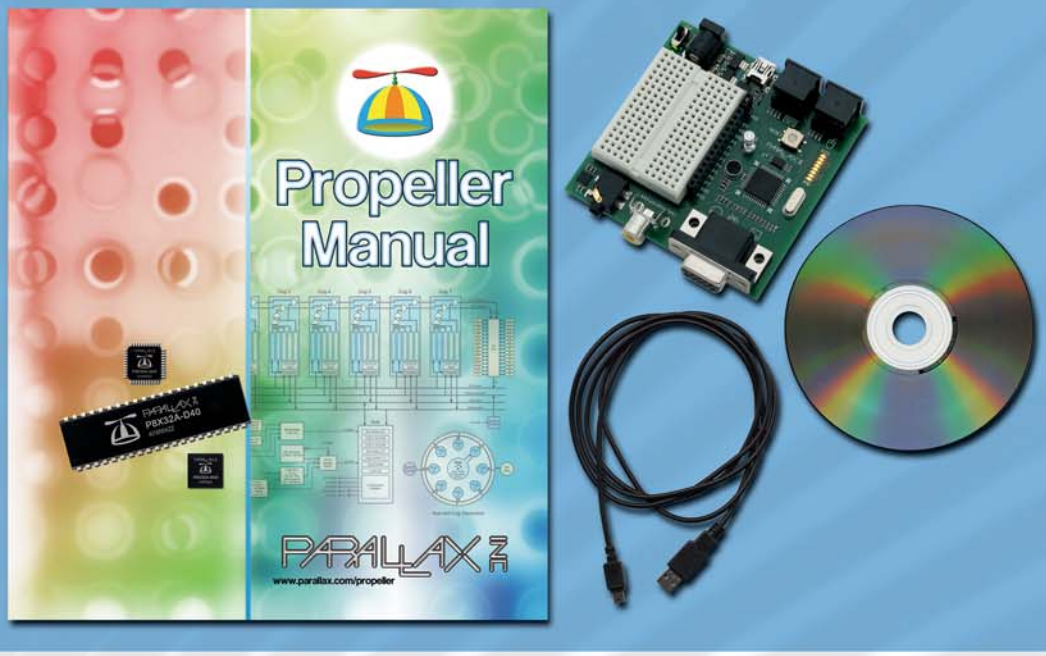
TechTools

www.tech-tools.com

(972) 272-9392 • sales@tech-tools.com

Copyright © 2006 TechTools • DigiView, FlexROM, EconoROM and QuickWriter are trademarks of TechTools • PICmicro is a registered trademark of Microchip Technology Inc.

Propeller™ *Starter Kit*



The new Propeller Starter Kit includes everything you need to get started with the Propeller microcontroller, including the Propeller Demo Board, the Propeller Manual, software on CD, and a USB cable. The Propeller Demo Board includes a built-in Propeller (P8X32A-Q44), EEPROM and 5 MHz crystal pre-wired to connectors for interfacing to devices such as a mouse, keyboard, TV or VGA monitor and speakers. This is all configured for immediate use by many objects from the Propeller Object Library (included with the Propeller Tool software). In just a few minutes you can see some of the impressive tasks the Propeller can perform.

Propeller Starter Kit (#32300; $149.95)

Order online at www.parallax.com or by calling the Parallax Sales Department toll-free at 888-512-1024 (Mon-Fri, 7 a.m. - 5 p.m., PT).

PARALLAX

www.**parallax**.com