

File Storage Tips (p. 51) • Programmable Robotics (p. 56) • The "8-Bits Story" Revisited (p. 64)

www.circuitcellar.com

# CIRCUIT CELLAR

THE MAGAZINE FOR COMPUTER APPLICATIONS

#224 March 2009

## ROBOTICS

Robot Navigation Control  
Subsystem

Wi-Fi-Enabled Mobile  
Surveillance

Vision-Guided Balancing  
Robot

Networked Timing  
System



\$5.95 U.S. (\$6.95 Canada)

# SECURE SERIAL-TO-ETHERNET SOLUTION



**Low-cost**



**32-bit Performance**



**SSH/SSL Secured**

## Need a custom solution?

Customize with the NetBurner SB70LC Development Kit for only \$99.

Customize any aspect of operation including web pages, data filtering, or custom network applications.

Kit includes: platform hardware, ANSI C/C++ compiler, TCP/IP stack, web server, e-mail protocols, RTOS, flash file system, Eclipse IDE, debugger, cables and power supply

Kit enables communication with peripherals that use: SD/MMC Flash Card (including SDHC), SPI, I<sup>2</sup>C, or the general purpose digital I/O interface

The NetBurner Security Suite Option includes: SSH v1, v2 and SSL support

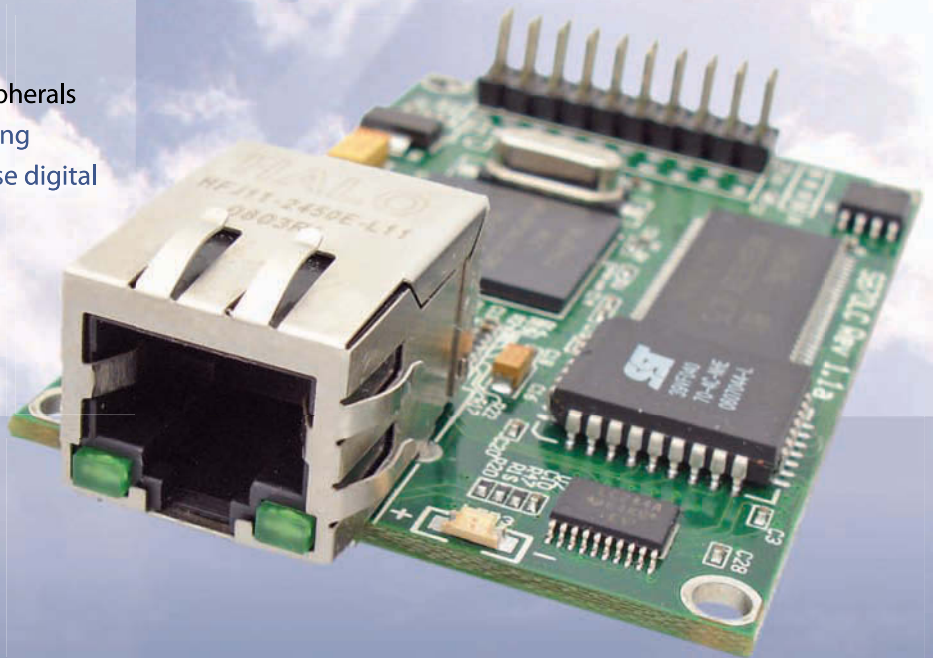
## The complete, **secure** hardware and software solution

- Simple Ethernet connectivity for serial devices
- Works out of the box - no programming is required
- Enable data encryption to prevent unauthorized monitoring
- Customize to suit any application with development kit

### Features:

- 10/100 Mbps Ethernet
- SSH/SSL/TCP/UDP modes
- DHCP/Static IP support
- Web-based configuration
- Two TTL serial ports

**SB70<sub>LC</sub>**  
**\$49<sup>00</sup>** Qty. 100



Board Part Number | SB70LC-100CR  
Development Kit Part Number | NNDK-SB70LC-KIT  
Information and Sales | sales@netburner.com  
Web | www.netburner.com  
Telephone | 1-800-695-6828



# 5 Competitive Advantages

## Overseas Manufacturing

Imagineering, Inc. enjoys the reputation of being one of the most experienced & successful offshore PCB suppliers.

## CAM USA

Our Illinois based DFM office has eight fully staffed CAD / CAM stations. Within hours of receipt of new Gerber files, our highly experienced DFM engineers conduct thorough and precise analyses.

## Quick-Turn Production

Imagineering offers small volume production in 5-6 days and medium to large volume production in 2-3 weeks.

## Overseas Manufacturing

## Shipping Logistics

With Imagineering there is no need to deal with multiple suppliers, language barriers, customs headaches, and shipping logistics. We do it all for you ..and deliver door-to-door

## Significant Price Saving

Our global buying power combined with the capabilities of our overseas manufacturers translate into tremendous savings to our customers.

Quick-Turn  
Production

CAM USA

Door to Door  
Delivery

Significant  
Price Saving

## Capabilities

- Up to 30 Layers
- Blind Buried Vias
- Di-Electric Thickness
- Impedance Control (TDR Tested)
- Plated Edge Holes
- Up to 6oz Copper
- 6 mil Laser Drill
- 3 mil line width/spacing
- Conductive Epoxy Filled Vias
- Aluminum Metal Core Boards
- ...and many others

ITAR, ISO 9001 : 2000

Over the past 5 years, 70,000 prototypes have been successfully delivered from overseas to over 5000 customers



**Imagineering Inc.**

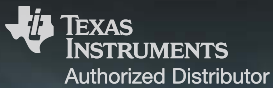
847-806-0003

[www.PCBnet.com](http://www.PCBnet.com)

email: [sales@PCBnet.com](mailto:sales@PCBnet.com)

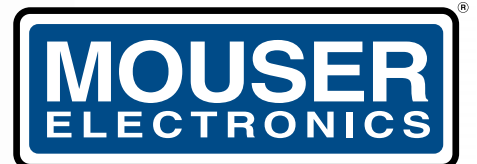
23 YEARS IN BUSINESS...AND STILL GOING STRONG

# The Newest Semiconductors



The ONLY New Catalog Every 90 Days

Experience Mouser's time-to-market advantage with no minimums and same-day shipping of the newest products from more than 366 leading suppliers.



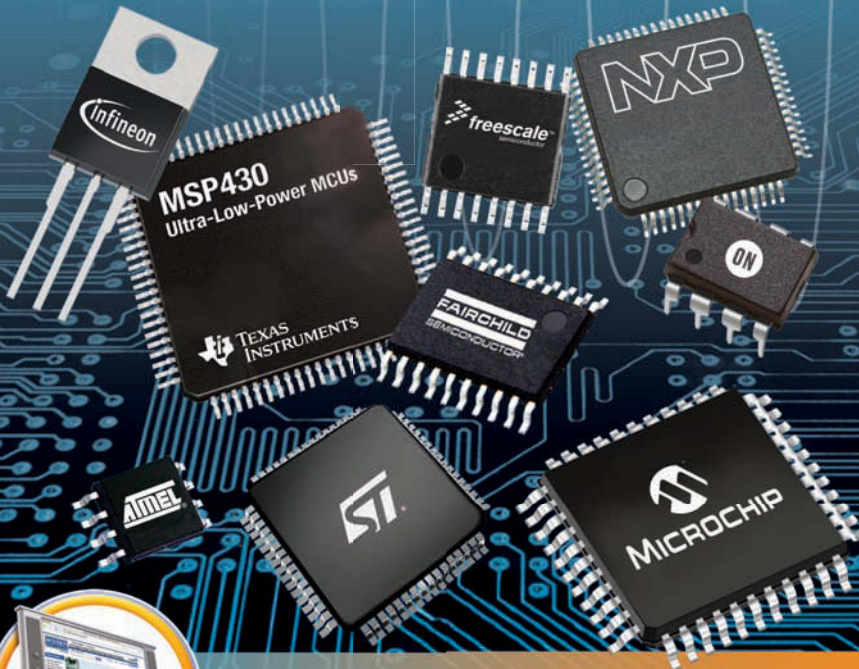
a tti company

The Newest Products For Your Newest Designs

(800) 346-6873

[www.mouser.com](http://www.mouser.com)

Over A Million Products Online



The brighter side of electronics

# Noritake

## Enhanced Replacement for LCD

- Character Highlight
- Character Magnification
- Increased Message Capability
- Multiple Interface Solutions

NEW Y-SERIES VFD



VFD



LCD

- Largest Viewing Angle
- International Font Tables
- Bright Display - 2000cd/m<sup>2</sup>
- 5 x 8 Characters with Descender
- Widest Temperature Range (- 40 to +85 °C)

RoHS  
ISO 9001  
ISO 14001

[www.noritake-elec.com/54](http://www.noritake-elec.com/54)

**itron**<sup>®</sup>

Noritake Co., Inc. 2635 Clearbrook Dr., Arlington Heights, IL 60005 phone 1-800-779-5846 e-mail [electronics@noritake.com](mailto:electronics@noritake.com)

Inventor of the Vacuum Fluorescent Display

## Design Evolution

As I've said here before, most quality new designs are the sum total of dozens, perhaps hundreds, of earlier projects, applications, and programs. When you see a project in this magazine, you can think of it as a single point in the long timeline of technological evolution. Looking ahead, you should consider each project, idea, and program described in these pages as a contribution to future projects. You and your fellow readers will take what you learn and put it to good use during your upcoming endeavors.

This month, we feature interesting robotics applications. These robots are the progeny of countless other robotics applications that are easy to find and study. Look no further than your stack of past *Circuit Cellar* magazines to read about many of the designs that influenced the development of the robots we use today. A few *Circuit Cellar* articles immediately come to mind: J. Bingham and L. Magnusson, "Inertial Rolling Robot" (*Circuit Cellar* 200, 2007); E. Leland, et al, "Robot Localization and Control" (*Circuit Cellar* 188, 2006); M. Chao and L. Ming, "GPS-GSM Mobile Navigator," (*Circuit Cellar* 151, 2003); M. Dvorsky, "Fighting Fire with Robots: How to Build a Mobile Robot Base," (*Circuit Cellar* 128, 2001); J. Stefan, "Navigating With GPS," (*Circuit Cellar* 123, 2000); B. Reynolds, "MicroBot: Programming Intel's 8749 for Robotic Control" (*Circuit Cellar* 92, 1998); I. Cyliax, "Robot Navigation Schemes," (*Circuit Cellar* 81, 1997); and C. McManis, "Sensing Obstacles with Mobile Robots," (*Circuit Cellar* 73, 1996).

In this issue, Guido Ottaviani contributes to the development of new robotics control systems. In the first part of his series titled "Robot Navigation and Control," he describes how to design a navigation control subsystem (p. 14).

Starting on page 22, Hanno Sander describes a novel vision-guided robotics application. His exciting balancing robot design was on display at the Embedded Systems Conference in San Jose, CA, last year. Perhaps you saw it in action?

In "Wireless Mobile Robotics," Scott Coppersmith presents a compact, Wi-Fi-enabled mobile robot (p. 41). He mounted a webcam on the top of the robot so it can transmit real-time images to a remote laptop.

In the final robotics-related article, Jeff Bachiochi introduces the topic of programmable robotics (p. 56). He describes how to upgrade and develop an existing robotics platform.

If you're interested in timing applications, Thomas Bereiter's article is right up your alley. In "Networked Timing," he explains how he built a timer with advanced planning tools (p. 31). The irrigation control system generates handy information such as zone activity.

Turn to page 51 for a review of the FAT file system. In this first part of the series, George Martin covers the process of opening files and performing operations.

Tom Cantrell closes the issue with an article about one of his favorite topics: 8-bit chips (p. 64). Although 32-bit chips get a lot of attention these days, Tom explains that 8-bit chips are alive and well.

Good luck starting your next project. Feel free to share your ideas and design experiences on the *Circuit Cellar* Discussion Board: <http://bbs.circuitcellar.com/phpBB2/>.

[cj@circuitcellar.com](mailto: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

Robert Lacoste

George Martin

Ed Nisley

## NEW PRODUCTS EDITOR

John Gorsky

## PROJECT EDITORS

Gary Bodley

Ken Davidson

David Tweed

## ASSOCIATE EDITOR

Jesse Smolin

## CHIEF FINANCIAL OFFICER

Jeannette Ciarcia

## MEDIA CONSULTANT

Dan Rodrigues

## CUSTOMER SERVICE

Debbie Lavioie

## CONTROLLER

Jeff Yanco

## ART DIRECTOR

KC Prescott

## GRAPHIC DESIGNERS

Grace Chen

Carey Penney

## STAFF ENGINEER

John Gorsky

## ADVERTISING

860.875.2199 • Fax: 860.871.0411 • [www.circuitcellar.com/advertise](http://www.circuitcellar.com/advertise)

## PUBLISHER

Sean Donnelly

Direct: 860.872.3064, Cell: 860.930.4326, E-mail: [sean@circuitcellar.com](mailto:sean@circuitcellar.com)

## ADVERTISING REPRESENTATIVE

Shannon Barraclough

Direct: 860.872.3064, E-mail: [shannon@circuitcellar.com](mailto:shannon@circuitcellar.com)

## ADVERTISING COORDINATOR

Valerie Luster

E-mail: [val.luster@circuitcellar.com](mailto:val.luster@circuitcellar.com)

Cover photography by Chris Rakoczy—Rakoczy Photography  
[www.rakoczyphoto.com](http://www.rakoczyphoto.com)

PRINTED IN THE UNITED STATES

## CONTACTS

### SUBSCRIPTIONS

**Information:** [www.circuitcellar.com/subscribe](http://www.circuitcellar.com/subscribe), E-mail: [subscribe@circuitcellar.com](mailto:subscribe@circuitcellar.com)

**Subscribe:** 800.269.6301, [www.circuitcellar.com/subscribe](http://www.circuitcellar.com/subscribe), Circuit Cellar Subscriptions, P.O. Box 5650, Hanover, NH 03755-5650

**Address Changes/Problems:** E-mail: [subscribe@circuitcellar.com](mailto:subscribe@circuitcellar.com)

### GENERAL INFORMATION

860.875.2199, Fax: 860.871.0411, E-mail: [info@circuitcellar.com](mailto:info@circuitcellar.com)

**Editorial Office:** Editor, Circuit Cellar, 4 Park St., Vernon, CT 06066, E-mail: [editor@circuitcellar.com](mailto:editor@circuitcellar.com)

**New Products:** New Products, Circuit Cellar, 4 Park St., Vernon, CT 06066, E-mail: [newproducts@circuitcellar.com](mailto:newproducts@circuitcellar.com)

### AUTHORIZED REPRINTS INFORMATION

860.875.2199, E-mail: [reprints@circuitcellar.com](mailto: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 © 2009 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.

LUMINARY MICRO®

# Redefining Microcontrollers: Connected. Versatile. Cost-effective.

We understand that it is  
all about the software, so  
we've changed the rules.

See how our connected, versatile  
and cost-effective Stellaris devices  
will help you change the rules  
in your market.



28-pin SOIC



48-pin LQFP



100-pin LQFP



108-pin BGA



64-pin LQFP

Learn more about how we have changed the rules today:  
[www.luminarymicro.com/newrules](http://www.luminarymicro.com/newrules)

STELLARIS®  
microcontrollers

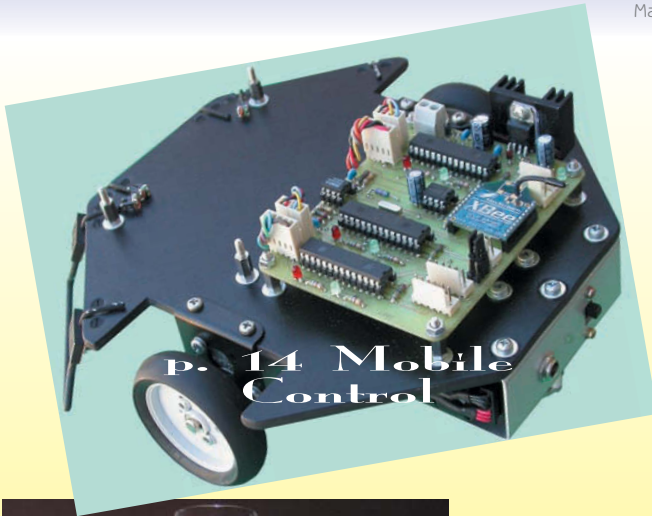


LUMINARY MICRO®

# INSIDE ISSUE

# 224

March 2009 • Robotics



p. 14 Mobile Control

**14** **Robot Navigation and Control (Part 1)**  
Construct a Navigation Control Subsystem  
*Guido Ottaviani*

**22** **Vision-Guided Robotics**  
A Next-Generation Balancing Robot  
*Hanno Sander*

**31** **Networked Timing**  
Build a Timer With Advanced Planning Tools  
*Thomas Bereiter*

**41** **Wireless Mobile Robotics**  
A Wi-Fi-Enabled System With a Mounted Webcam  
*Scott Coppersmith*



p. 22 Balancing Robot Design



p. 56 Work from the Inside Out

**51** **LESSONS FROM THE TRENCHES**  
FAT File System Review (Part 1)  
Open Files and Perform Operations  
*George Martin*

**56** **FROM THE BENCH**  
Programmable Robotics (Part 1)  
Build on an Existing Robot Platform  
*Jeff Bachiochi*

**64** **SILICON UPDATE**  
A Really Simple Plan  
The "8-Bits" Saga Continues  
*Tom Cantrell*

**TASK MANAGER** **4**  
Design Evolution  
*C. J. Abate*

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

**TEST YOUR EQ** **13**

**CROSSWORD** **71**

**INDEX OF ADVERTISERS** **79**  
225 Preview

**PRIORITY INTERRUPT** **80**  
Cloud Computing  
*Steve Ciarcia*



# Hammer Down Your Power Consumption with picoPower™!



## THE Performance Choice of Lowest-Power Microcontrollers



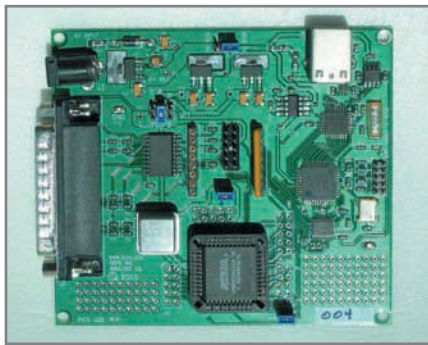
Performance and power consumption have always been key elements in the development of AVR® microcontrollers. Today's increasing use of battery and signal line powered applications makes power consumption criteria more important than ever. To meet the tough requirements of modern microcontrollers, Atmel® has combined more than ten years of low power research and development into picoPower technology.

picoPower enables tinyAVR®, megaAVR® and XMEGA™ microcontrollers to achieve the industry's lowest power consumption. Why be satisfied with microamps when you can have nanoamps? With Atmel MCUs today's embedded designers get systems using a mere 650 nA running a real-time clock (RTC) and only 100 nA in sleep mode. Combined with several other innovative techniques, picoPower microcontrollers help you reduce your applications power consumption without compromising system performance!

Visit our website to learn how picoPower can help you hammer down the power consumption of your next designs. PLUS, get a chance to apply for a **free AVR design kit!**

<http://www.atmel.com/picopower/>

**ATMEL**  
Everywhere You Are®



## CPLD DEVELOPMENT KIT

The new **picoUSB** development kit for Altera's MAX3000 and MAX7000 families of CPLDs enables you to economically learn Altera CPLD design. With its USB interface, picoUSB in-system programming can be easily managed from your PC. It is an ideal solution for introductory digital logic design courses.

The picoUSB uses a 44-pin PLCC socket to support the MAX3000 and 7000 devices. With the free Quartus II Web Edition design software package and the built-in programmers, you will be programming your own designs into the device quickly. Power is provided either through the USB interface or with an external 9-V, 2.5-mm adapter. The development kit brings your I/O pins out to headers and the voltage is selectable for 5-, 3.3-, or 2.5-V logic levels.

The picoUSB comes complete with a user's manual, reference design, and example HDL code. The picoUSB starts at **\$145** per kit. Volume and university pricing are also available.

**Colorado Electronic Product Design, Inc.**  
[www.cepd.com](http://www.cepd.com)

## POCKET-SIZED LINUX PC WITH GPS AND GPRS

The **AarLogic C10/3** is a breadboard that contains a complete Linux PC on a surface of just 104 mm x 63 mm. A quad-band GPRS module and a SiRF3 GPS module are also on board, as well as interfaces for USB, RS-232, and Ethernet components. An SD card reader enables the trouble-free expansion of the 4 Mb of RAM that comes standard.

The heart of the PC is two ARM processors that are responsible on one hand for the GSM component and on the other for applications executable under Embedded Linux. The processor module, including the GSM component, is also available for purchase separately. Despite its small surface area, roughly the size of a matchbox, its 192-pin socket provides a wide array of connectivity options. Aside from keyboards, digital cameras, and reading devices, this also includes WLAN, Bluetooth, and GPS components.

The AarLogic C10/3 costs approximately **\$590**.

**Round Solutions**  
[www.roundsolutions.com](http://www.roundsolutions.com)



"Producing prototypes  
on-the-fly  
allows me to be more  
Creative"

Electrical engineers agree: with a Protomat S-Series prototyping machine at your side, you'll arrive at the best solutions, fast. These highly accurate benchtop PCB milling machines eliminate bread-boarding and allow you to create real, repeatable test circuits—including plated vias—in minutes, not days.

- Declare your independence from board houses
- Affordable, entry-level price tag
- The best milling speed, resolution, and accuracy in the industry
- Single-sided, double-sided, and multilayered machining without hazardous chemicals
- Optional vacuum table and autosearch camera for layer alignment

## ProtoMat® S-Series PCB Milling Machines



**LPKF**®

Laser & Electronics

For complete details visit:  
[www.lpkfusa.com](http://www.lpkfusa.com)  
or call:  
**1-800-345-LPKF**

## AVR AND PIC DEVELOPMENT KITS

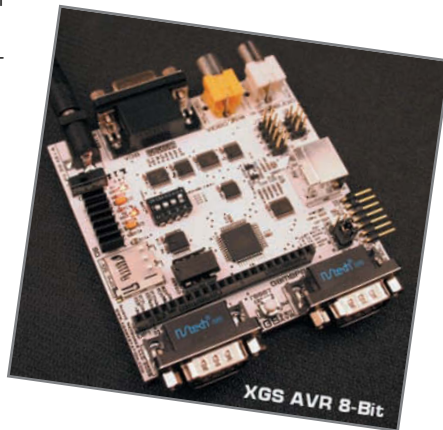
The **XGS AVR 8-bit** and **XGS PIC 16-bit** are new development kits that make learning and development easy. The XGS AVR 8-bit is based on the Atmel ATmega644 AVR processor and is a highly integrated development kit for exploring ATmega AVR processors. The XGS PIC 16-Bit is based on Microchip Technology's new 16-bit PIC24 processor. Both kits are designed to be serious development kits for schools, students, engineers, and anyone interested in learning programming.

The XGS kits give designers a fun way to learn AVR and PIC programming in the context of graphics, audio, and simple game development. So, instead of blinking LEDs and displaying digits on a seven-segment display, designers will develop graphics applications that control the VGA and NTSC screen to learn C and ASM.

Each kit comes with a 350-plus-page printed manual covering hardware, software, and numerous programming tutorials; a programmer and USB cable; a 9-V power supply; an AV cable; a game controller; and a PC serial port to XGS header converter. Also included is a DVD with numerous examples and a complete driver library including graphics, sound, keyboard, SD card, serial communications, mechatronics, and more. Bonus materials on the DVD include electronic copies of numerous game development and electronics gaming hardware books.

The XGS AVR 8-Bit costs **\$139.99** and the XGS PIC 16-Bit costs **\$159.99**.

**Nurve Networks**  
[www.xgamestation.com](http://www.xgamestation.com)



## WIRELESS ENVIRONMENTAL SENSING KIT

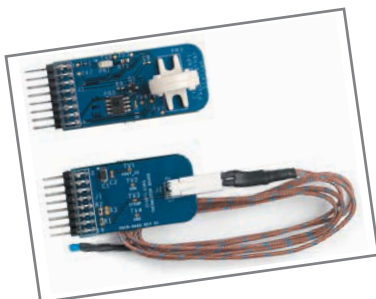
The **CY3271-EXP1 Environmental Sensing Kit** is the first of many future expansion offerings for the recently introduced PSoC First Touch Starter Kit with CyFi Low-Power RF. The Environmental Sensing Kit includes two boards: a weather station expansion board—which includes sensors for atmospheric pressure, humidity, temperature, and ambient light (all of which can be operated and monitored simultaneously)—along with a pigtail thermistor board for measuring temperature in remote locations.


The new sensing kit is easy to use right out of the box. You can use the "Sense and Control Dashboard" software included in the new FirstTouch Starter Kit to quickly set up and monitor a wired or wireless sensor network through the intuitive visual dashboard. The combination of the Environmental Sensing Kit and the dashboard enable data logging, data aggregation, alarms, and sensor calibration for nearly any type of sensor. In addition, the kit supports custom expansion for designers who want to develop their own applications.

The new CyFi Low-Power RF solution delivers the industry's leading combination of reliable connectivity, power efficiency, and superior range. It is supported by the flexible and easy-to-use PSoC programmable system-on-chip. The PSoC reduces the complexity of quantifying a sensor's signal by integrating the amplification, filtering, and ADC stages into a single device. Additionally, the PSoC's analog MUX allows multiple sensors to be connected concurrently.

The CY3271-EXP1 Environmental Sensing Kit costs **\$99.99** and the CY3271 PSoC FirstTouch Starter Kit costs **\$69.95**.

**Cypress Semiconductor Corp.**  
[www.cypress.com](http://www.cypress.com)





Software Trials Available

## JTAG (J-Link™)

+++ ARM7/9 Cortex M3 +++

- USB to JTAG
- Fast 720kb/s Download Speed
- Serial Wire Debug (SWD) Support
- Multicore Debugging Support
- Auto JTAG Speed Recognition

The J-Link can be coupled with a number of available software modules to fit your application needs.

**J-Flash** is a stand-alone application used with the J-Link to program internal and external flash devices.


**J-Link RDI** permits the use of the J-Link with an RDI compliant debugger.

**J-Link GDB Server** is a remote server for GDB.


**J-Link Flash Download** is a module used to download your program into flash even if your debugger does not have a flash loader.

**J-Link Flash Breakpoint** permits you to set an unlimited number of software breakpoints while debugging in flash.

**J-Link SDK** is a standard DLL that extends the full functionality of the J-Link to your proprietary application.



We also offer a JTAG isolator which can be used to offer electrical isolation between your target hardware and the J-Link. This is essential when the development tools are not connected to the same ground as the application. It is also useful to protect the development tools from electrical spikes that often occur in some applications, such as motor control applications.



**Special Offer**  
Includes the J-Link, J-Link GDB Server, and the J-Link Flash Download.  
[www.segger-us.com/ncu.html](http://www.segger-us.com/ncu.html)

**J-Link Non-Commercial (NCU) Bundle**

[www.segger.com](http://www.segger.com)

NPN

March 2009 – Issue 224

9

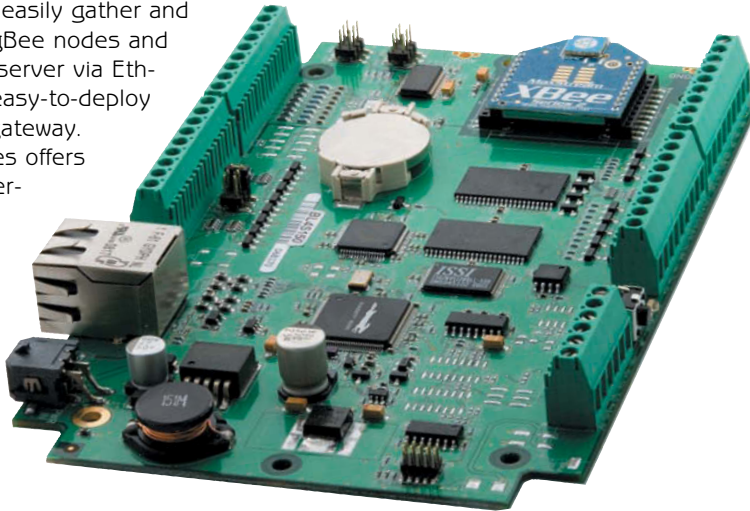
## NEW FAMILY OF WIRELESS SINGLE-BOARD COMPUTERS

The BL45100 and BL45200 are SBCs that feature either Wi-Fi or ZigBee connectivity, a microprocessor, memory, and abundant I/O that enable easy deployment of wireless nodes for industrial, commercial, and medical applications. This unique combination of control, I/O, and connectivity makes it easy for engineers to add wireless connectivity and control to devices like vision systems, wireless industrial control systems, printing systems, automatic meter-reading devices, industrial ventilation systems, and HVAC systems. They also enable design engineers to easily gather and control data from ZigBee nodes and uplink the data to a server via Ethernet, providing an easy-to-deploy ZigBee-to-Ethernet gateway.

The BL45100 series offers both ZigBee and Ethernet connectivity and provides embedded design engineers a simple and straightforward approach for machine control and data acquisition without

the burden of cables and wiring harnesses. It enables design engineers to deploy ZigBee nodes at various control points and connect the nodes wirelessly to the BL45100 board. The BL45100 SBC can then gather and collate the data from the ZigBee nodes and uplink it to a server via Ethernet. The high-performance BL45200 series delivers higher clock speeds and twice the I/O as the BL45100 series and offers a choice of Wi-Fi or ZigBee connectivity. It is ideal for applications that require significant digital and analog I/O such as data logging, instrument reading, and controlling motors, relays, and solenoids.

The BL45100 and BL45200 development kits are available that include the essential hardware and software tools. The BL45100 kits cost **\$199**. The BL45200 kits cost **\$299**. The BL45100 SBCs are available in 500-plus quantities starting at **\$98**. The BL45200 SBCs are available in 500-plus quantities starting at **\$176**.



**Rabbit Semiconductor, Inc.**  
[www.rabbit.com](http://www.rabbit.com)

NPN



**We add value to PCBs when others just sell it.**

- One Stop Manufacturing Service
- Free Electronics Components
- Free Prototyping Assembly
- Professional Consultant



### Designing Service

3D Enclosure Designing Virtual Assembly PCB Design



Pcb



Assembly



Quick Prototype



Component



Fpc



Keypad

Pcbs Assembly Quick Prototype

Components Fpcs Keypads



[www.EzPCB.com](http://www.EzPCB.com)

Email: [sales@ezpcb.com](mailto:sales@ezpcb.com)

## EMBEDDED LAN MODULE

**Nano LANReach** is an embedded LAN module that quickly and easily connects any embedded device to 10/100BaseT LANs with minimal programming. Based on the CO2144 chip, Nano LANReach offers plug-and-play serial-to-LAN functionality enabling immediate and full-featured LAN connectivity by connecting an Ethernet cable to the onboard RJ-45 connector. In addition, the module includes a full suite of Internet protocols, applications, and security engines.

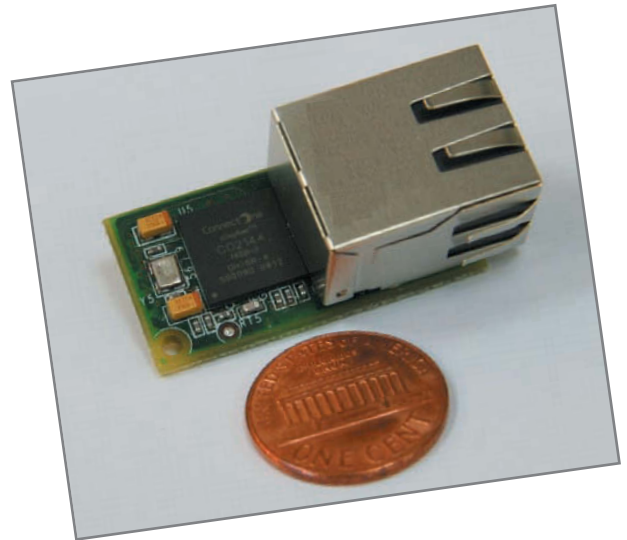
Nano LANReach's firmware supports two modes of operation. The first is serial-to-LAN bridge mode, which enables transparent bridging of serial data over a LAN using the module's high-speed UART. The module supports TCP, UDP, or SSL tunneling of serial data. The second is full Internet controller mode, which enables simple microcontrollers to use the module's rich protocol and application capabilities to perform complex Internet operations, such as e-mail, FTP, SSL, embedded web server, and others. Internet controller mode can be used with any of the hardware interfaces.

Nano LANReach includes USB, SPI, and fast UART interfaces for easy integration into existing or new designs. In addition to SerialNET, it supports 10 simultaneous TCP/UDP sockets: two listening TCP sockets; SMTP, MIME, POP3, FTP, Telnet, and HTTP/HTTPS clients; and an HTTP/HTTPS embedded web server with a web site for the host application and one for configuring the module. Nano LANReach also supports AES-128/256, SHA-128/192/256, 3DES, the

SSL3/TLS1 protocol for a secure client socket session, and a secure FTP session.

The Nano LANReach starts at **\$38**. The **\$160** II-EVB-363ML evaluation board provides an easy environment for developing applications and testing.

**Connect One**  
[www.connectone.com](http://www.connectone.com)



NPN

## System on Module Internet Appliance Engine

- Atmel ARM9 400Mhz CPU
- 6 Serial Ports & 2 SPIs
- Up to 60 Digital GPIOs
- 10/100 BaseT Ethernet
- SODIMM Bus Expansion
- SSC/I2S Audio Interface
- Linux with Eclipse IDE
- Real Time Clock Calendar
- SD/MMC Flash Card Interface
- Up to 512 MB Flash & 64 MB RAM
- 2 USB 2.0 Host Ports & 1 Device Port
- 4 10-Bit A/Ds & 6 16-Bit Timer/Counters



The SoM-9G20 uses the same small SODIMM form-factor utilized by other EMAC SoM modules, and is the ideal processor engine for your next design. All of the ARM9 processor core is included on this tiny board including: Flash, Memory, Serial Ports, Ethernet, I2S Audio Interface, PWMs, Timer/Counters, A/D, digital I/O lines, Clock/Calendar, and more. Like other modules in EMAC's SoM product line, the SoM-9G20 is designed to plug into a custom or off-the-shelf Carrier board containing all the connectors and any additional I/O components that may be required. The SoM approach provides the flexibility of a fully customized product at a greatly reduced cost. Single unit pricing starts at \$150.

<http://www.emacinc.com/som/som9G20.htm>

Since 1985  
OVER  
24  
YEARS OF  
SINGLE BOARD  
SOLUTIONS

**EMAC, inc.**  
EQUIPMENT MONITOR AND CONTROL

Phone: ( 618 ) 529-4525 • Fax: (618) 457-0110 • Web: [www.emacinc.com](http://www.emacinc.com)

## FRONT PANELS & ENCLOSURES

Customized front panels can be easily designed with our free software  
**Front Panel Designer**

- Cost-effective prototypes and production runs
- Wide range of materials or customization of provided material
- Automatic price calculation
- Fabrication in 1, 3 or 7 days

Sample price:  
\$43.78 plus S&H



**FRONT PANEL EXPRESS**

[www.frontpanelexpress.com](http://www.frontpanelexpress.com)  
 (206) 768-0602

## VIBRANT THREE-COLOR DISPLAYS READABLE UP TO 600'

The **Plant Floor Marquee (PFM)** is a series of highly visible, multicolor LED displays optimized for communicating critical machine status and production data instantly across the plant floor. The largest PFM is easily readable at distances up to 600' (182 m). The PFM series features a graphical three-color display—available in four sizes—that can convey production targets, machine faults, and other custom messages.

The PFM takes advantage of the powerful features of a G3 series HMI, Data Station Plus or Modular Controller as a host device to display the text and graphics on the PFM. Red Lion Controls's host devices and free Crimson software enable the PFM to receive message data from virtually any device on the factory floor, supporting over 170 different protocols and the ability to send and receive data from seven or more device types simultaneously.

The PFM using a Red Lion host offers several additional features for enhanced usability and unprecedented flexibility, including vibrant green, yellow, and red LEDs. It has four sizes (ranging from 26" x 6.6" at 80 x 16 resolution to 39.3" x 22.5" at 120 x 64 resolution) and can send and receive data from seven or more device types simultaneously. It also incorporates bar graphs, custom symbols, and other graphics to display production trends and goals. It presents data from a single device or uses custom calculations to display optimized production data from multiple devices. They also feature an addressable RS-485 port to support multidrop wiring.

Please contact Red Lion Controls for pricing.

**Red Lion Controls, Inc.**  
[www.redlion.net](http://www.redlion.net)



## 802.11B/G WLAN ACCESS POINT MODULE

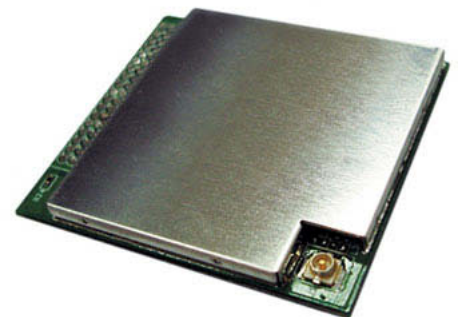
The **WIZ610wi** is an 802.11b/g WLAN access point module in a pin-head-type package. Serial to WLAN can be easily implemented with any system that has a UART interface. With the WIZ610wi, development time is reduced because WLAN driver porting is not required. Also, the module guarantees excellent stability as a result of independent operation.

The module provides Ethernet-to-wireless bridging,

enabling a wired network device to have a wireless communication interface. It also offers a serial interface for "serial to wireless" applications and offers an AP/Station/Gateway mode. The WIZ610wi offers a 54-Mbps data rate and a maximum 20-Mbps effective data transmission rate. The module includes a full-featured network protocol stack and a built-in web server for configuration via a standard web browser.

The WIZ610wi (including antenna) costs **\$41.90** each. The WIZ610wi-EVB evaluation board costs **\$99**.

**WIZnet, Inc.**  
[www.ewiznet.com](http://www.ewiznet.com)



**We Listen. Think. And Create.**

Distributed I/O    Serial I/O    **Digital I/O**    Industrial Computing    HMI

**SeaDAC USB modules are the fastest, most reliable way to connect I/O to any computer.**

**SeaDAC USB Modules Offer:**

- Optically Isolated Inputs, Reed Relay Outputs, Form C Relay Outputs, Digital/Analog Combo
- Status Indicator LEDs for Communication, Fault, & Status
- Field Removable Terminal Block Connectors
- High-Retention USB Type B Connector
- Rugged Plastic Tabletop Enclosure
- Extended Temperature Option Available

**FOCUS**  
 On Success  
 Call Today!

**SEALEVEL**  
[sealevel.com](http://sealevel.com) > [sales@sealevel.com](mailto:sales@sealevel.com) > 864.843.8067

**Problem 1**—What is a “shell” program? Give some examples.

**Problem 2**—What exactly does a shell do?

**Problem 3**—Explain the concept of a shell “environment.” What, for example, is PATH?

**Problem 4**—What are some of the key ways in which GUI shells are different from text shells?

*David Tweed contributed the four problems and answers in this issue.*

**What's your EQ?**—The answers are posted at [www.circuitcellar.com/eq/](http://www.circuitcellar.com/eq/)  
You may contact the quizmasters at [eq@circuitcellar.com](mailto:eq@circuitcellar.com)

## Got Serial, Need Network?

**Bluetooth Qty 1 \$145**

**Ethernet Qty 1 \$99**

**Wireless Qty 1 \$199**

**Volume Discounts Available**

**gridconnect™**  
[www.gridconnect.com](http://www.gridconnect.com)  
+1 800 975-4743

## RF Specialists

**RF Modules**  
From Part 15 to Part 90 Compliant  
Narrow Band FM, UHF Multi-Channel

**GSM/GPRS**  
M2M Solutions  
GSM/GPRS modules and modem series

**Industrial Bluetooth**  
OEM, Modules, Wireless Device Servers, RS-232  
Long range options, low cost

**Data Loggers**  
Stand Alone and  
Wireless Mesh Networking Logger

**GPS**  
OEM Modules and USB ZigBee Sticks,  
Mesh Networks

**ZigBee Pro**  
OEM Modules and USB ZigBee Sticks,  
Mesh Networks

**Wi-Fi**  
**RS232 / 422 / 485 to Wi-Fi Adapter**  
Connect Data Acquisition Equipment through Serial Port to Wi-Fi network

**LEMOS INTERNATIONAL**  
[www.lemosint.com](http://www.lemosint.com)  
866.345.3667  
[sales@lemosint.com](mailto:sales@lemosint.com)

# Robot Navigation and Control (Part 1)

## Construct a Navigation Control Subsystem

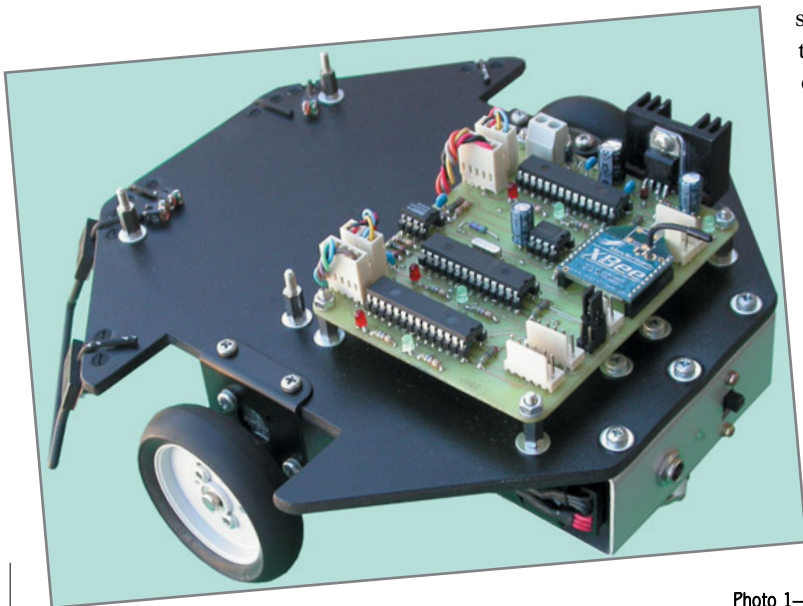
Guido built a navigation control subsystem for an autonomous differential steering explorer robot. In the first part of this article series, he describes a robotic platform that drives motors and controls an H-bridge. Guido also presents a communication system that remotely manages the robot.

During the early stages of my “electronic childhood,” I dreamt of building an autonomous robot. But such a project was too difficult and expensive back then. Now it’s a lot easier with powerful, inexpensive hardware and a development system that can run on a standard computer. Such devices are readily available on the Internet.

I recently made my dream come true by building an autonomous robot. In the first part of this article series, I will describe the navigation control subsystem that I designed for a differential steering explorer robot (see [Photo 1](#)). The “dsNavCon” system, as I call it, features a Microchip Technology dsPIC30F4012 motor controller and a general-purpose dsPIC30F3013.

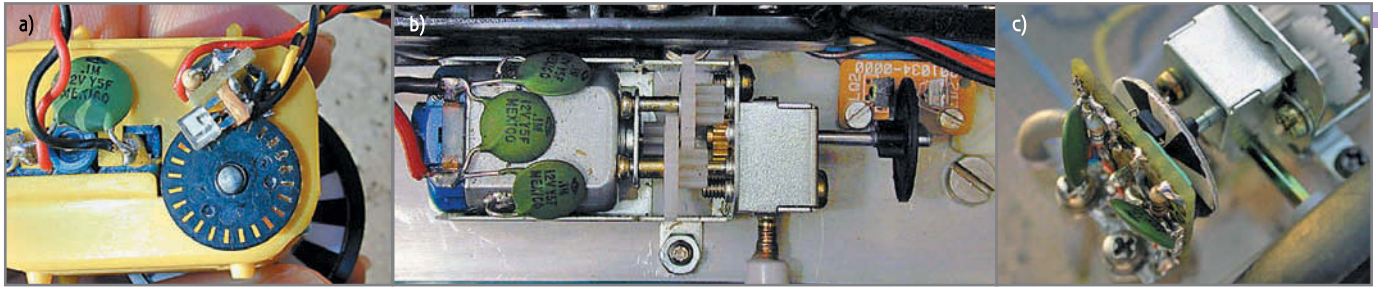
### EXPERIMENTATION

I started visiting some amateur explorer robot competitions a few years ago. I was disappointed with the robots I encountered. Many seemed to move around at random. Others seemed to repeat the same path several times. Fortunately, while searching the Internet, I found Johann Borenstein’s technical report, “Where Am I?: Sensors and Methods for Mobile Robot Positioning,”<sup>[1]</sup> and the SR04 robot by



**Photo 1**—This is the entire navigation control subsystem (as described in Figure 1). It is installed on the Rino robotic platform.





**Photo 2**—These are my first experiments with homemade encoders. **a**—This is a hacked mechanical mouse. **b**—I used a different kind of geared motor with mouse parts. **c**—I used photoreflectors on a printed wheel.

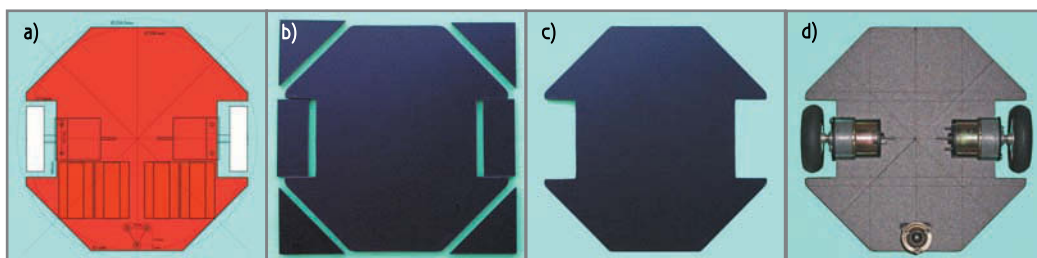
David Anderson.<sup>[2]</sup> I was impressed by odometry and dead reckoning. Most of the robots built by amateurs are based on a differential steering system: a couple of driving wheels and a caster. Knowing the space covered by each wheel periodically with enough precision enables you to calculate the position coordinates of the robot at any given moment.

As you can see in [Photo 2](#), I tested different quadrature encoders. Any dead-reckoning navigation system is affected by cumulative error. The measuring precision must be high to ensure a small error circle after a long path. After some good results with homemade encoders, I used something better: a couple of 12-V 200-RPM geared motors connected to a couple of 300-count-per-revolution (CPR) quadrature encoders. These parts are available at many online robotics shops.

I designed my simple robotic platform with easy-to-find components and parts that didn't require professional tools or equipment or special skills. I purposely developed a design that is properly sized for several different categories of robotics projects (e.g., exploring robots, line-following robots, and collecting robots). I named it "Rino," which rhymes with Robottino, or "little robot" in Italian (see [Photo 3](#)).

## BASIC PRINCIPLES

The 300-CPR encoders are connected to the motors' axles. Because the wheels spin at up to 200 RPM and the gear reduction ratio is 30:1, the axles can spin at 6,000 RPM. To catch all of the pulses generated by the encoders in a 4× decoding method (120 kHz), I needed dedicated hardware for each encoder. After experimenting with the Microchip PIC18F2431 motor control microcontroller, it was time to upgrade to the dsPIC30F digital signal controller series (DSC).



**Photo 3**—This is a sequence of the mobile platform starting from a square-cut standard piece of expanded PVC. This kind of material is available on the 'Net. Already cut to 200 mm x 200 mm x 5 mm, it needs just a cutter and a ruler to be shaped in such an easy way. An octagonal-based robot is as easy to drive as a circular one, but it's much easier to build.

A dsPIC30F4012 motor controller for each motor is perfect for controlling wheel position and speed and for performing odometry. Data provided by the two motor controllers is collected by a dsPIC30F3013. This general-purpose DSC has enough power to get data and perform some trigonometry to calculate position coordinates and store data related to the path covered to obtain a map of the field, all at a high rate.

This brings me to the dsPIC-based navigation control board, the dsNavCon. This board is designed to be part of a more complex system. In a complete explorer robot, other boards will control sound, light, and gas sensors, as well as bumpers and ultrasonic range finders for locating targets and avoiding obstacles. A behavior board will decide how to act to reach the goal.

As a stand-alone board, the dsNavCon can be used for a simple line-following robot or various other robotics applications. There is still plenty of free program memory in the supervisor dsPIC to add code for such tasks. With minor changes in software (or none at all), it can also be used by itself for a remote-controlled vehicle (using the bidirectional RF modem with some kind of smart remote control). This remote control can send complex commands such as "move FWD 1 m," "turn 15° left," "run FWD at 50 cm/s," "go to X, Y coordinates," or something similar.

## SUBSYSTEM

The navigation control subsystem is shown in [Figure 1](#). A detailed schematic diagram and pictures of every board, as well as the complete project done with Cadsoft's Eagle, are available on the *Circuit Cellar* FTP site.

The subsystem includes the dsNavCon and an STMicroelectronics L298-based dual H-bridge board for controlling the geared 12-V motors (Hsiang Neng DC Gear Motor Manufacturing's HN-GH12-1634TR). The circuit of this H-bridge is a classic application of an L298 IC, as you can see in its datasheet and application note. Motion feedback comes from a couple of 300-CPR quadrature

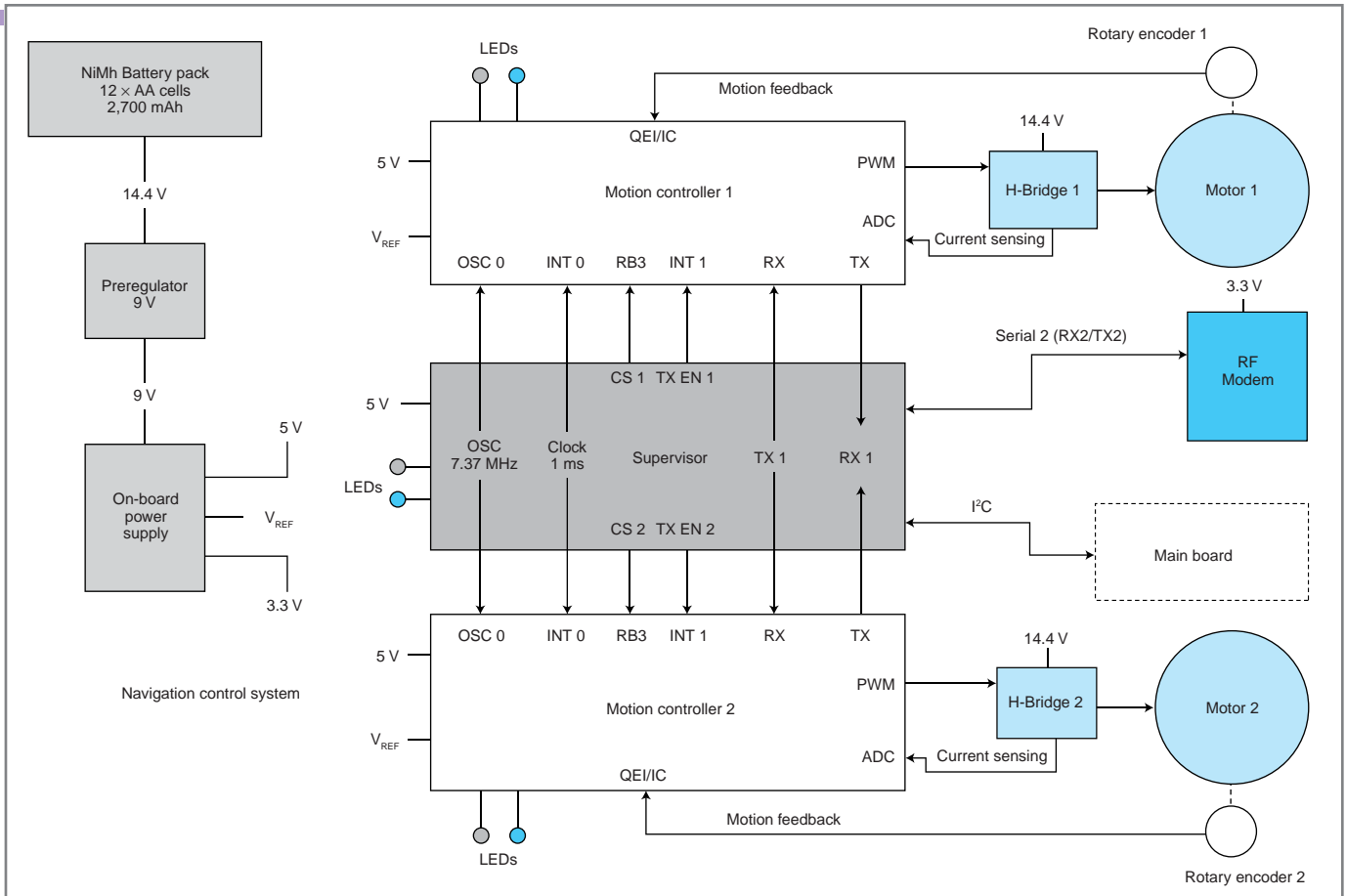


Figure 1—This is the navigation control subsystem.

encoders (US Digital E4P-300-079-HT miniature optical kit encoder). A 12-AA NiMH cell battery pack, with a nominal voltage of 14.4 V and 2,700 mAh, supplies the power (see Photo 4). This ensures the correct voltage for the motors after the loss of the H-bridge. A couple of LM7809 regulators drop down the 14.4 V to 9 V for a logic board's power supply. The voltage regulators are decoupled with a capacitor-input (Pi) filter on each one. This system reduces interference from the motors and enables the use of a smaller heatsink for the 5-V regulators on each individual board. The power supply on the dsNavCon board also provides 3.3 V for the ZigBee RF modem (MaxStream's (now Digi International) XBee module) and 2.7-V reference voltage for the motor controllers' ADC. This converter is used to read the motors' current through a 0.27- $\Omega$  shunt resistor on each H-bridge and a couple of op-amps with a gain of 10 on the dsNav-Con board.

The supervisor communicates with the behavior board of the robot through an I<sup>2</sup>C bus and with a remote PC via a ZigBee RF modem for telemetry (UART2-RX2/TX2). The supervisor drives both motor controllers (MCs) through UART1 (RX1/TX1) communication, sending commands and reading information (space, speed, and motor current). The motor controllers have no oscillator hardware; the supervisor provides them with a 7.3728-MHz signal obtained by an Output Compare simple PWM peripheral. A

1-ms timing signal is also provided by the supervisor to synchronize every operation. The supervisor controls motor controllers communications and other operations through the use of chip-select I/O signals.

## DESIGN EVOLUTION

New components are always hitting the market. My first PID program was developed on a Microchip PIC16F877 microcontroller with a quadrature encoder interface performed in

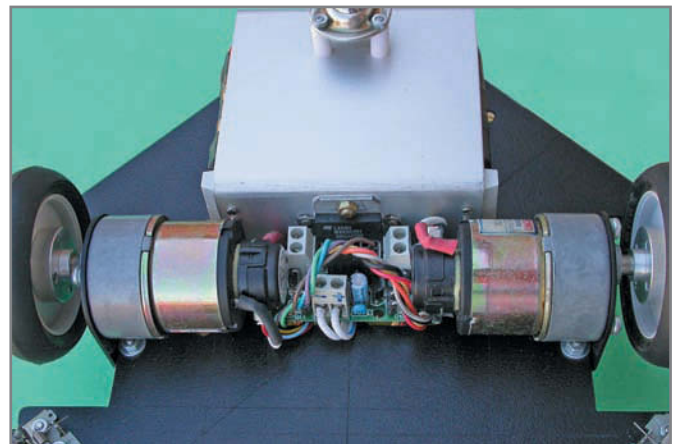


Photo 4—These are the motors, the encoders, and the H-bridges. The metal box at the top contains a battery pack, 9-V preregulators, and PI filters. It also acts as a heatsink.

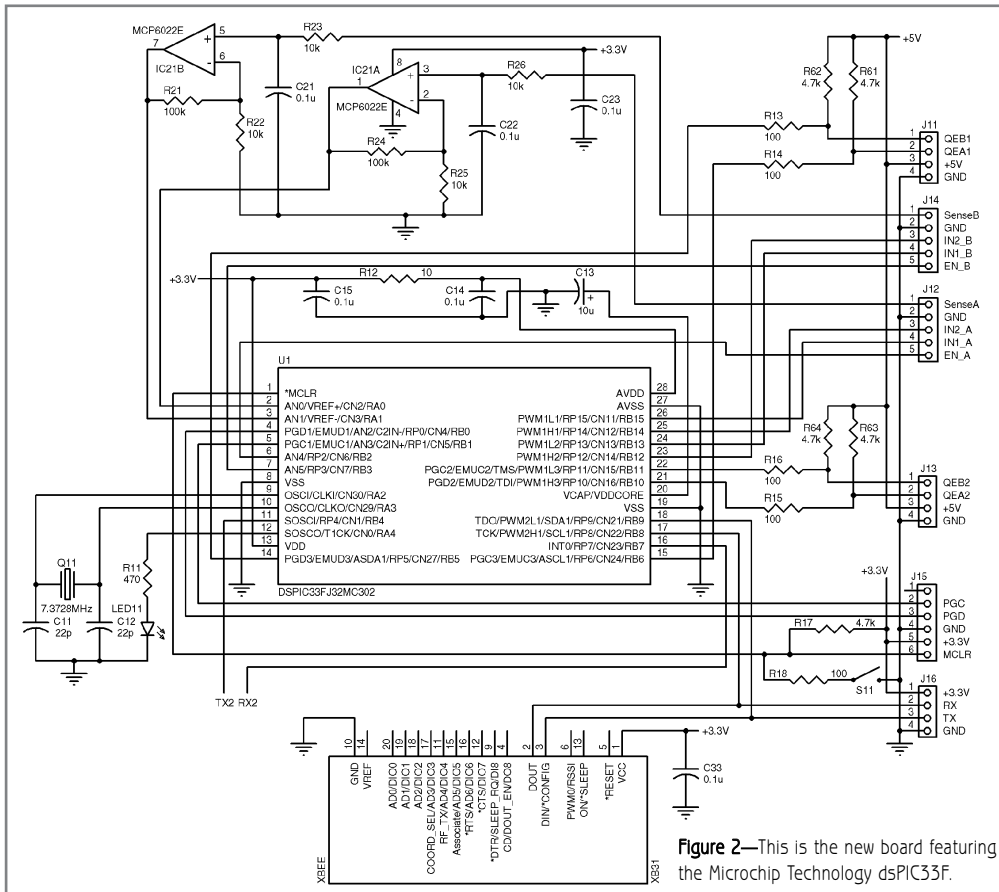


Figure 2—This is the new board featuring the Microchip Technology dsPIC33F.

much simpler. There is no need for high-speed communication between the supervisor and motor controllers to exchange navigation parameters. Every process is simple to synchronize because it's on the same chip.

The peripheral pin select capability of the dsPIC33F series further simplifies the PCB, enabling an internal connection of peripherals and greater flexibility (see Figure 2). For example, the IC capture modules are internally connected to QEI pins, saving two precious pins. Moreover, the COMM-1 port used for XBee connection can be software switched from UART to I<sup>2</sup>C or SPI communication with no hardware modifications. Note that every one of the 28 pins is used on this board.

software. When I started porting the software on a PIC18F2431, the hardware QEI interface looked like the perfect solution. The math capability of the dsPIC30F series arose as a useful option for trigonometric calculi needed for odometry. There are dual-in-line versions of this DSC and they are faster.

When the board and the programs were almost completed, Microchip brought out a new, powerful 28-pin SPDIP in the dsPIC33F series for both motor controller (MC) and general-purpose (GP) versions. They are significantly faster than the dsPIC30F, they have a lot more available program memory and RAM (useful for field mapping), they require less power (good for a battery-operated robot), and their DMA capabilities simplify many I/O operations. Most importantly, these are the first Microchip motor controllers with two QEIs on the same chip. Let's start a new port again!

The logical block diagram is similar to Figure 1 for the previous board, but the hardware and software are

THE ORIGINAL SINCE 1994

# PCB-POOL.COM

Beta LAYOUT

## Specializing in Quickturn Proto's

- Internet pioneers with 15 years experience
- Instant online Quotations & Ordering
- From Singlesided to 6 layers ML
- Leadtimes from 48 hrs
- Full DRC included on all orders
- High Quality prototypes at LOW cost's

Simply send your layout files and order online

[www.pcb-pool.com](http://www.pcb-pool.com)

TollFree USA: 1877 3908541 Email: sales@pcb-pool.com

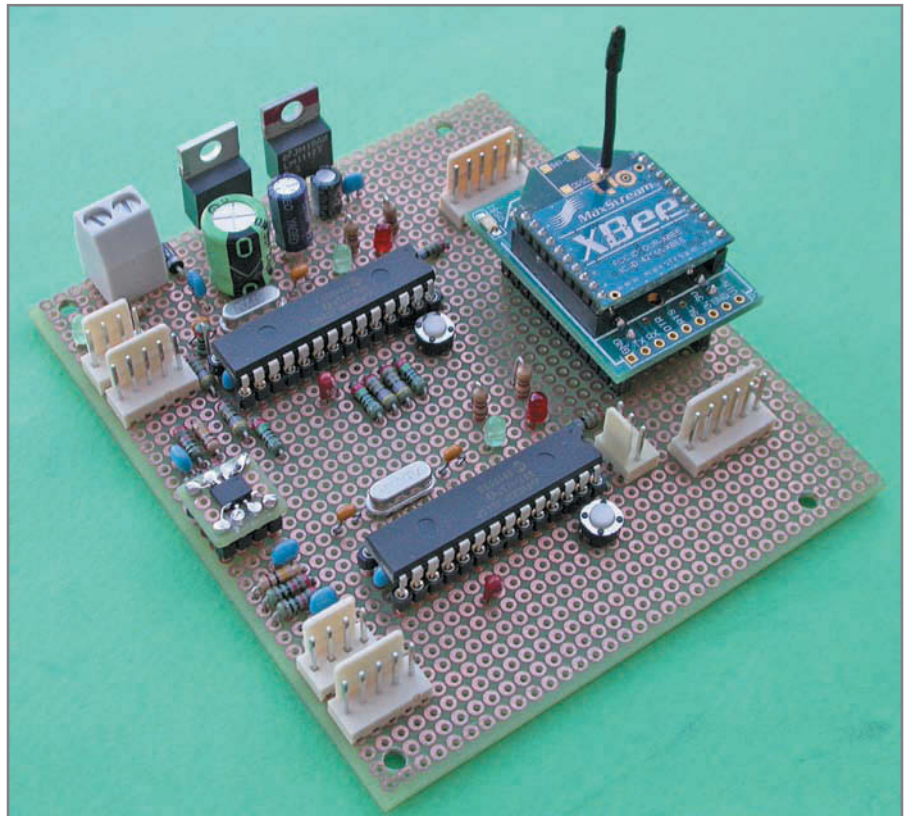
The number of components and connections is dramatically reduced. On a board that is the same size as the previous one, there is enough room for a second GP series DSC that will manage all of the robot's sensors. **Photo 5** shows the development board used for porting.

At the end of the software porting process, I confirmed my first impression: one dsPIC33FJ64MC302 is powerful enough to manage all of the navigation tasks. I had to take care just for the high rate of the input capture interrupts. But with the timing chosen for the different PIDs, the program spends more than 80% of its time idling in the main loop. I'll cover this in more detail in the second part of this article series.

## TELEMETRY

Any kind of closed-loop control (i.e., PID or other) requires a fine setup to achieve its purpose. If you really want to ask "Where am I?" any parameter of the program must be fine-tuned. Several groups of parameters must be tested before finding the right sequence. Believe me, the most boring method is to change values in code, recompile, and burn the flash memory of the dsPIC over and over with the hex file. You absolutely need an I/O system to read and write numbers to and from the program.

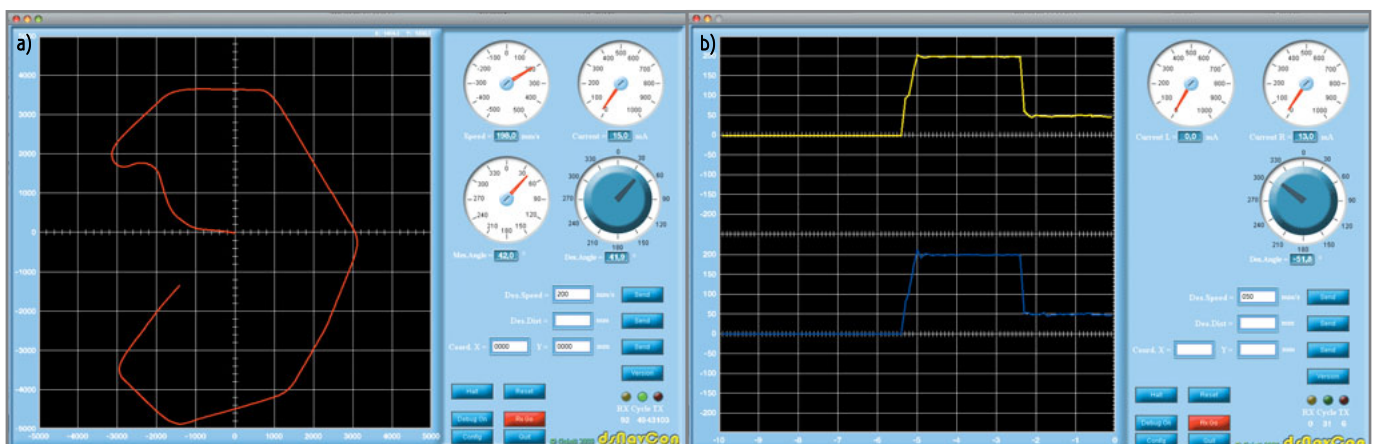
It's common to install a simple LCD on a robot. A standard display (e.g., 4 × 20) compatible with the Hitachi 44780 protocol is inexpensive and easy to



**Photo 5**—The new board based on the Microchip dsPIC33F is a lot simpler—in terms of both hardware and software—than the previous one.

program. But it's not easy to use. It cannot show a lot of information at the same time and it is slow. Plus, it is attached to a moving vehicle. Excluding graphic displays for the same reason and because they are expensive and hard to program, a good solution would be a wireless system that exchanges data with a computer, on which it's easy to display the numbers, even in graphical format, and to store large amount of bytes.

Serial communication is a good way to connect a wireless system. Two serial ports on the DSC and a simple protocol, such as the one I will describe later, can be used. Using a couple of XBee modules in transparent mode, serial communication can be made wireless without the need for any other protocol or special configuration. On the dsPIC side, just RX and TX pins are needed. An affordable adapter is available to



**Photo 6a**—The navigation panel shows the robot's current position and the path. The knob and input fields enable you to control navigation. **b**—The details panel shows the present value for each motor's current and a graph with the trend of the speed values for both wheels.

connect an XBee module to a PC via USB. They are versatile, reliable, and easy to use, with just one problem. For some reason, the MaxStream developers decided to install an unusual contact strip with a 2-mm metric pitch instead of a standard 100-mil pitch. Fortunately, someone designed a board that adapts the XBee pins to a standard strip connector, adding an LDO 3.3-V regulator and some signaling LEDs in a footprint that is a little bit bigger than XBee itself. Refer to the Resources section for more information about the XBee simple board and XBee USB board adapters.

Because you have the physical layer and the protocol for communication, you need an application layer that shows the results on a computer screen. A good tool would be the National Instruments LabVIEW (graphical development) or LabWindows/CVI (C language compiler and IDE), with all of the easy-to-use widgets and primitives to develop a control panel or a virtual instrument. But its price is “professional,” which means it is costly. Still, I recommend looking at the full working timed demo.

I used my own widgets and instruments with a simpler development system. I wrote the console with the Processing open-source programming language and environment, and with the aid of the Interfascia graphical user interface library. (Refer to the Resources section at the end of this article for more information.) Processing is a simple, multiplatform environment that produces Java executables for Windows, Mac OS X, and Linux operating systems. It fits my requirements for software sharing perfectly.

Three panels make up the console. The main panel (or navigation panel) controls the robot’s movements. You can set values with a knob (for angle) or input fields for speed, distance, or target coordinates (see Photo 6a). Instruments return the actual values for mean speed, total current for both motors, and orientation. The graph shows the current position and the path.

The configuration panel contains the input fields for all of the constant parameters used to set up the program. Values are stored in the DSC’s flash memory once they are sent. They can be saved in a file on the computer side.

The details panel shows the speed and current values for each motor (see Photo 6b). The graphical representation of speed is essential to set up Speed PID K parameters because it reveals how the motors respond to

variations in load or required velocity.

## CALIBRATION PROCEDURES

Once the right communication procedure and display of data is running, you can start the setup. There are many online documents about the proper calibration procedures for the PID constants, both theoretical and practical. Most of them are valid. A simple method that is applicable in this situation is explained in Microchip’s code example CE019.<sup>[3]</sup>

**Altium**

**NOW WITH DYNAMIC LIVE 3D PCB DESIGN**

**Streamline the entire electronics design process within a single unified solution.**

- > Rapid prototyping using a live, interactive, reconfigurable hardware development system
- > A smooth path from concept exploration through live 3D PCB design and out to successful manufacture
- > Easy connection to company-wide systems

**Seeing is believing. Visit Altium’s next generation electronics design solutions today.**

[www.altium.com/view-demo](http://www.altium.com/view-demo)

There are four PID algorithms in this software that control the navigation. Two for speed of the wheels, one for orientation, and one to control the distance from the target. The four PIDs are independent, so they can be set up separately. First of all, you can take care of the speed PID of one motor controller, changing the  $K_p$ ,  $K_i$ , and  $K_d$  values in sequence until the motor runs smoothly, responding quickly to the commands and without overshoot. The other

motor will probably run fine with the same constants. After the speed can be regulated accurately, the right  $K$  values for angle PID first and distance PID later can be found with the same logic.

The axle size and wheel diameter can be measured on the robot with a caliper, but they must be fine-tuned to make the vehicle go straight, turning the right angle and running the right distance. You can achieve the first tuning by comparing the

## RTC COMPETITION

As design engineers, we all need a goal. By "goal" I mean, a target that justifies the amount of time we spend debugging and trying to make our circuits work like they did in the simulations.

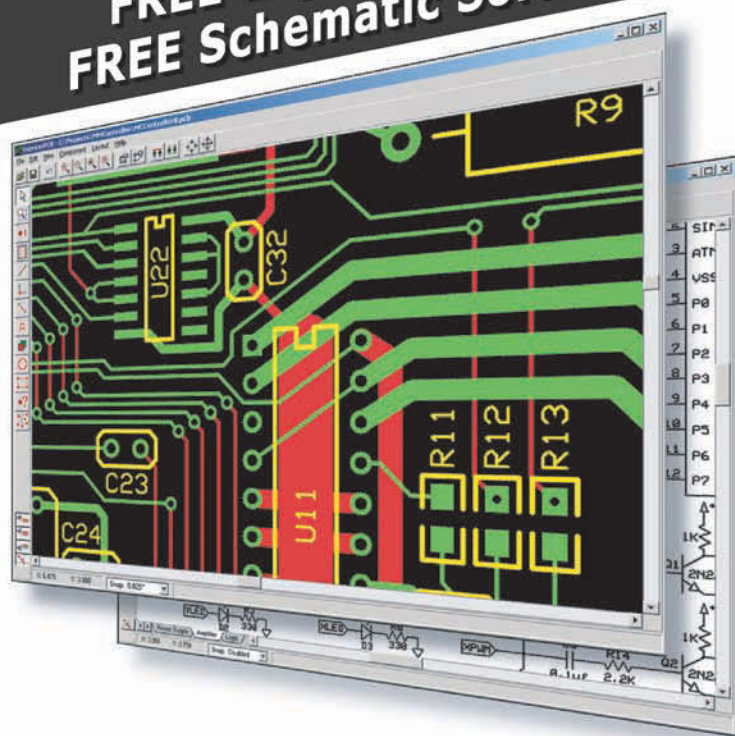
In an effort to get more people interested in the goal of building robots, a bunch of us members of the Roboteck Internet Discussion Group recently spent some time designing a competition. As mechanical insects walked around our meeting table (among all the pizza, beer, and red wine), we drafted a set of rules for the Robo Tolomeo Cup (RTC) competition. We named it the Robo Tolomeo Cup in order to reference the great Greek cartographer. Remember: navigation is key for any mobile robotics system.

The rules are simple. Each robot must start from point A, navigate to point B at least 10 m away, and then return to point A. This must be done with pure dead reckoning—without any external reference—while avoiding a few convex obstacles just so the path is not a straight line. The score is inversely proportional to the distance from the marked returning point at the end of the session.

A Rino-like system, the dsNav-Con board, and the theory behind it are all an entrant needs to build a robot for this kind of competition. The first experimental competition is scheduled to take place this year. We are intentionally keeping it low-profile without sponsors or prizes. The only prize is the satisfaction of finishing a project. If there are enough participants who submit interesting designs, we may plan another, more involved, competition.

The RTC could be considered an indoor version of—or stepping stone toward—a larger event such as the Robo-Magellan outdoor robotics competition. Who knows what the future will bring?

**\$51<sup>For 3</sup> PCBs**  
**FREE Layout Software!**  
**FREE Schematic Software!**



- 01 DOWNLOAD our free CAD software
- 02 DESIGN your two or four layer PC board
- 03 SEND us your design with just a click
- 04 RECEIVE top quality boards in just days

**expresspcb.com**

position displayed on console with the position really measured on field. A more accurate method is UMBmark, which was developed at the University of Michigan.<sup>[4]</sup>

## UP AND RUNNING

The robotic platform is considered up and running when the motors are spinning the wheels, the H-bridge is driving the motors, the board that controls the H-bridge is functioning (in the dsPIC30F or dsPIC33F versions), and the communication system needed for remote management is complete. When the system is ready and you have a goal (e.g., the RTC Competition) you can start working.

But wait. What else do you need? The software!

Next month, I will describe the software you need on the board to navigate the robot. I will cover how to control the speed with PID closed-loop control. You will also learn how to use the encoder information to determine the robot's position with dead reckoning by odometry. (Be prepared for some math.) Lastly, I'll cover the overall software architecture that glues all of the pieces together. 📦

*Author's note: Along with other members of the Roboteck Internet Discussion Group, I helped create a design competition for robotics enthusiasts. For more information about the Robo Tolomeo Cup (RTC), refer to the RTC Competition sidebar.*

*Guido Ottaviani (guido@guiott.com) has worked with electronics and ham radios for years. After working as an analog and digital developer for an Italian communications company for several years, Guido became a system integrator and then a technical manager for a company that develops and manages graphic, prepress, and press systems and technologies for a large Italian sports newspaper and magazine publisher. A few years ago, he dusted off his scope and soldering iron and started making autonomous robots. Guido is currently an active member in a few Italian robotics groups, where he shares his experiences with other electronics addicts and evangelizes amateur robotics.*

## PROJECT FILES

To download code, go to [ftp://ftp.circuitcellar.com/pub/Circuit\\_Cellar/2009/224](ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2009/224).

## REFERENCES

- [1] J. Borenstein, H. R. Everett, and L. Feng, "Where am I?: Sensors and Methods for Mobile Robot Positioning," Technical Report, University of Michigan, 1996.
- [2] D. Anderson, "SR04 Robot," Roy M. Huffington Department of Earth Sciences, Southern Methodist University, [www.geology.smu.edu/~dpa-www/robots/sr04/](http://www.geology.smu.edu/~dpa-www/robots/sr04/).
- [3] Microchip Technology, Inc., "dsPIC30F Code Examples: CE019—Proportional Integral Derivative (PID) controllers & closed-loop control," 2005, [www.microchip.com/stellent/idcplg?IdcService=SS\\_GET\\_PAGE&nodeId=2620](http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=2620).
- [4] J. Borenstein and L. Feng, "UMBmark: A Method for Measuring, Comparing, and Correcting Odometry Errors in Mobile Robots," 1994, [www-personal.umich.edu/~johannb/umbmark.htm](http://www-personal.umich.edu/~johannb/umbmark.htm).

## RESOURCES

- B. Fry and C. Reas, "Processing web site," [www.processing.org](http://www.processing.org).
- Interfascia graphical user interface library, <http://superstable.net/interfascia>.
- Droids SAS, "XBee Simple Board 990.001," [www.droids.it/data\\_sheets/990.001%20datasheet.pdf](http://www.droids.it/data_sheets/990.001%20datasheet.pdf).
- , "XBee USB Board," [www.droids.it/data\\_sheets/990.002%20datasheet.pdf](http://www.droids.it/data_sheets/990.002%20datasheet.pdf).
- G. Ottaviani, [www.guiott.com/Rino/index.html](http://www.guiott.com/Rino/index.html).
- Roboteck Discussion Group, <http://it.groups.yahoo.com/group/roboteck/> (Italian) or [http://groups.yahoo.com/group/roboteck\\_int/](http://groups.yahoo.com/group/roboteck_int/) (English)
- STMicroelectronics, "Application Note: Applications of Monolithic Bridge Drivers," AN240/1288, 1995.
- , "L298: Dual Full-Bridge Driver," 2000.

## SOURCES

### Eagle Software

CadSoft Computer | [www.cadsoftusa.com](http://www.cadsoftusa.com)

### HN-GH12-1634TR Motor

Hsiang Neng DC Gear Motor Manufacturing Corp. | [www.hsiangnengmotors.com.tw](http://www.hsiangnengmotors.com.tw)

### XBee Module

Digi International, Inc. | [www.digi.com](http://www.digi.com)

### PIC16F877 Microcontroller, PIC18F2431 microcontroller, dsPIC30F3013 digital signal controller, dsPIC30F4012 motor controller, and dsPIC33FJ64MC802 microcontroller

Microchip Technology, Inc. | [www.microchip.com](http://www.microchip.com)

### LabVIEW and LabWindows/CVI

National Instruments Corp. | [www.ni.com/labview](http://www.ni.com/labview)

### E4P-300-079-HT Miniature optical kit encoder

US Digital | [www.usdigital.com](http://www.usdigital.com)

# Vision-Guided Robotics

## A Next-Generation Balancing Robot

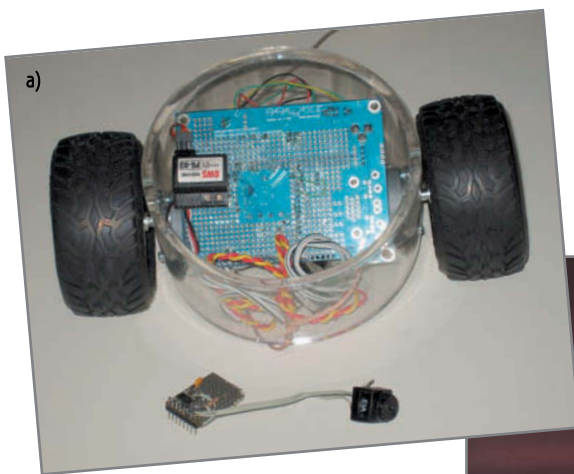
Are you interested in building a sophisticated, vision-guided, balancing robot that can interact with its environment? Hanno shows you how to tackle this project with a Parallax Propeller, a handy design kit, and an inexpensive camera.

It's time to build the next generation of robots. With today's technology, our robots should be tall enough to look us in the eye and interact with us through sight. Two years ago, I started experimenting with the latest microprocessor from Parallax, a camera, and a vision. In this article, I'll explain how to integrate vision technology in a project. As I describe the balancing robot design you see in [Photo 1](#), I'll cover parallel processing, visual debugging, vision algorithms, and robot control with computer vision.

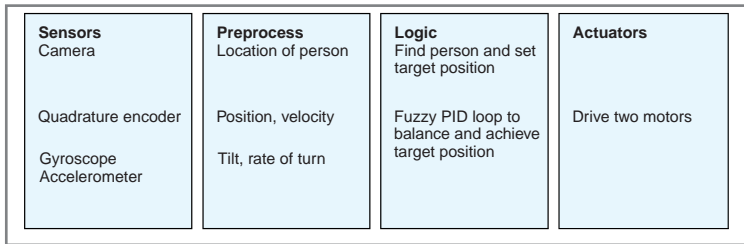
### PARALLAX PROPELLER

I started my project after my dad gave me a Parallax Propeller for Christmas. Parallax is best known for its Boe-Bot robot and BASIC Stamps, but its Propeller chip is quickly becoming popular with engineers and hobbyists because of its power and simplicity. Its eight identical processors (cogs) share common resources, such as global memory and an I/O port, but each can run its own program. It was perfect for my application because I needed to sample many different sensors at different rates, perform intensive filter calculations,

**Photo 1a**—This is the base of the DanceBot with wheels mounted on motors with quadrature encoders, the logic board powered by the Parallax Propeller, and the miniature camera with an ADC module. **b**—This autonomous balancing robot uses vision to interact with users. Here it's getting ready to balance a champagne flute for a month.







**Figure 1**—The DanceBot gets information from its environment through its sensors: a camera, a quadrature encoder, a gyroscope, and an accelerometer. It processes this data to find its dance partner, current position, and tilt. Fuzzy logic is used to balance and to maintain a set distance from its partner by driving the wheel motors.

process captured video, and control motors. I focused on parts of the problem and later integrated everything by assigning different algorithms to their own cogs.

The Propeller has eight 32-bit processors running at up to 80 MHz. It has shared global resources, including 32 KB of RAM, 32 KB of ROM, and 32 I/O pins. It has dedicated resources per processor, including 2 KB of RAM and two general counters, and video output. The Propeller operates at 3.3 VDC (each pin can sink up to 40 mA). It is available on Parallax's web site for \$12.99 per chip and \$29.99 per ProtoBoard.

## ViewPort

Parallax offers a free tool to load programs written in assembly or a high-level language called Spin to the Propeller. This is fine for getting started writing simple programs, but I quickly determined that I needed a more powerful debugging tool to develop and configure my vision-powered balancing robot—an application I now call ViewPort.

I started by dedicating one of the eight cogs to continuously share data stored in the Propeller's memory with a PC application. This enabled me to monitor and change variables while the other seven cogs ran at full speed. When I added a module that sampled all 32 I/O pins at 80 MHz, other designers became interested in using my application to debug their integration code—and ViewPort was born. Since then, I've added other capabilities to the ViewPort application to turn it into a complete debugging package. You can download a free 30-day trial of ViewPort at <http://mydancebot.com>.

## THE DanceBot

After watching friends demonstrate their iRobot Roomba robotic vacuum cleaners at a party, I wondered if I could build a balancing robot that could dance—not just with me, but with anyone in any environment. I wanted to build a robot that could dance with people and seem almost human.

Balancing robots make a great platform for mobile robots. They are highly maneuverable, have great traction, and move more smoothly and naturally than other designs. They can turn on a dime, navigate precisely, and are a pleasure to watch while they keep their balance. Unfortunately, building a robot that balances and maintains

position robustly in any environment is not easy.

The first lesson I learned was that unlike the inverted pendulum problem, a true balancing robot requires two control loops: one control loop to keep the robot from falling and another to keep the robot from losing its position. This combination also lets you move it programmatically. A significant milestone for building a balancing robot involves taking a simple step, accelerating to a set speed, travelling, and then decelerating to a stop. The DanceBot uses what's known as a "hybrid fuzzy logic cascading PID controller" to precisely carry out this and other advanced

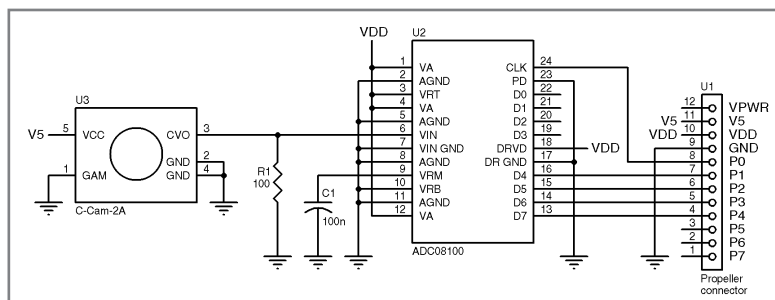
maneuvers. In the algorithm, the inputs to the PID controllers are first processed by a fuzzy logic engine to make the control algorithm more robust and easier to tune. The PID controllers, which correct the error between a measured variable and its setpoint by calculating a corrective action, are arranged in a cascade with the output of one used as the setpoint in the second.

Second, while it's possible to determine tilt by optically measuring the distance to the floor, this technique isn't robust. The DanceBot measures rate of turn using a ceramic gyroscope and integrates this signal to calculate tilt. Fusing the calculated tilt value with measurements from an accelerometer with a Kalman filter yields an accurate tilt reading with no drift. This combination lets the DanceBot stay balanced in any environment.

The DanceBot is controlled like a car: it requires two channels of information (see Figure 1). Channel 1, speed, controls how fast the robot should travel. Channel 2, turn rate, controls how quickly the robot should turn about its own axis. The DanceBot manages the speeds of its two motors to stay balanced and to achieve the position orientation and velocity goals given by its higher level planner. Unlike a car, the robot is capable of turning in place. At first, I controlled my robot with a remote control, but I quickly realized that it would be much more fun if it could interact with others as well—just by watching what they were doing. The first step to guide the robot with vision was to build a frame grabber.

## FRAME GRABBER

The DanceBot's vision is controlled by a small grayscale Electronics123.com C-Cam-2A miniature video camera. It is just 16 × 16 × 16 mm, uses less than 100 mW, and costs less



**Figure 2**—This is the frame grabber hardware. The C-Cam-2A outputs an NTSC composite signal in pin 3. This is digitized by the ADC08100 whose output D4, D7 is fed to the Parallax Propeller.

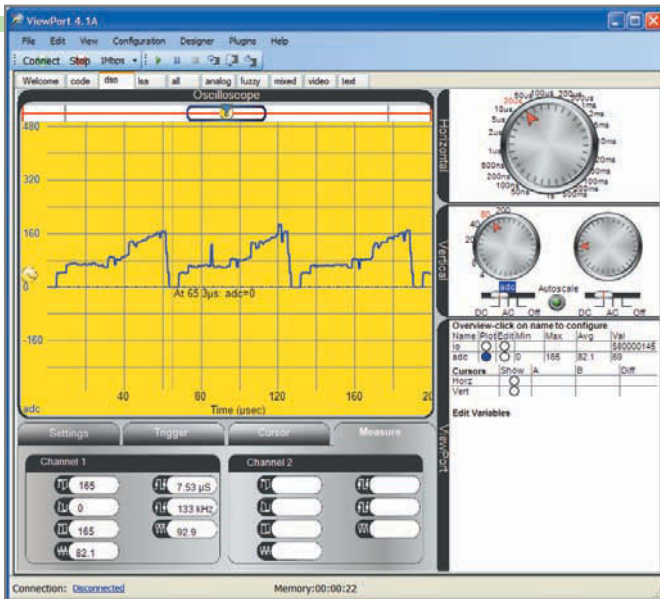


Photo 2—ViewPort shows raw NTSC signal from the camera, as digitized by the ADC.

than \$20. It has five pins, three of which provide ground and 5-V power, a gamma mode, and an output. The output signal consists of a 1-V<sub>pp</sub> composite video signal when terminated with a 75-Ω resistor to ground. To watch the camera's output, you can simply plug it into the composite input of your TV. It's that simple! Understanding what the camera sees is a bit harder, so I'll take it one step at a time.

First, you have to digitize the analog signal. To sample slower waveforms with the Propeller, you would typically use delta-sigma modulation with a capacitor and a resistor. But because you need to resolve the individual pixels in a frame, you need a faster solution.

The ADC08100 is a 20- to 100-Msps, 8-bit ADC. With a clock signal, it will output the digital equivalent of its input voltage on its eight digital outputs. We'll use one of the Propeller's 16 hardware counters to clock the ADC at 10 MHz and read the result from the Propeller's I/O port (see Figure 2).

At this point, your robot is ready to take its first peak at the world—one scan line at a time (see Photo 2).

Listing 1 is a short program that uses ViewPort to trigger and display the NTSC waveform generated by the camera.

The program starts by configuring the Propeller's clock to run at 80 MHz and including the three objects you need. The vp commands register is a component that will quickly sample the state of the I/O port and configure the ViewPort interface. Finally, the Freq.Synth call generates a 10-MHz clock to drive the ADC. Photo 2 shows the oscilloscope with a timescale of 10 μs/division. The waveform represents a horizontal trigger followed by a color burst and 50 μs of data. A pixel's brightness is proportional to the signal's value.

To complete your frame grabber object, your algorithm must detect the horizontal and vertical sync marks and then compress the pixel data into memory. Vertical and horizontal sync marks differ in the amount of time the signal stays at the lowest level. After detecting a vertical sync

**FlashPro430**  
**FlashPro-CC**  
**FlashPro2000**  
**GangPro430**  
**GangPro-CC**

USB Flash Programmers for Texas Instruments' MCUs  
**MSP430, Chipcon CCxx, C2000 DSPs**

**Reliable and the fastest programmer on the market.**  
**Perfect for production usage.**

- \* can assign unique serial number
- \* up to eight programmers can be connected to one PC and program target devices simultaneously

One PC and 8 programmers

**Elprotronic**  
 Incorporated  
[www.elprotronic.com](http://www.elprotronic.com)

**WIRELESS MADE SIMPLE®**  
 BRING YOUR PRODUCT QUICKLY AND LEGALLY TO MARKET

**RF Modules** Add **INSTANT** wireless analog / digital capability to your product.

- Low-Cost TX, RX & TRX Modules
- Multi-Channel Modules
- Long-Range Modules

**OEM Products** FCC PRE-CERTIFIED & ready to customize for your application.

- Handheld TXs
- Keyfob TXs
- Function Modules

**Feature Products** A closer look at Linx innovation

- LOW-COST • LONG-RANGE TRANSMITTER**
  - Direct serial interface
  - Low power consumption
  - PLL-synthesized architecture
  - RSSI and power-down functions
  - Compact surface-mount package
  - No external RF components (except antenna)
- REMOTE CONTROL TRANSCODER IC**
  - Up to 8 inputs
  - Bi-directional control
  - Transmitter ID output
  - Automatic confirmation
  - Secure 2<sup>31</sup> possible addresses
  - Latched and/or momentary outputs

**LINX TECHNOLOGIES**  
 WIRELESS MADE SIMPLE

**800-736-6677**  
 159 Ort Lane · Merlin, OR 97532  
[www.linxtechnologies.com](http://www.linxtechnologies.com)

**Listing 1**—This Spin program sets the Propeller's clock to run at 80 MHz and includes several objects: "Conduit" to graph the ADC's waveform via ViewPort on the PC, "QuickSample" to sample the Propeller's I/O pins, and "Synth" to generate the 10-MHz clock signal for the ADC. Running this on the Propeller enables you to display the NTSC waveform, as sampled by your frame grabber on the PC.

```

CON
    _clkmode      = xtall + pll16x
    _xinfreq      = 5_000_000

OBJ
    vp :          "Conduit" 'transfers data to/from PC
    qs :          "QuickSample" 'samples INA up to 80Mhz
    Freq :        "Synth"

pub demoADC[a,frame[1600+6]] 'frame stores 1600 samples+configuration
    vp.register(qs.sampleINA(@frame,1 ))
    vp.config(string("var:io,adc(decode=io[0..7])"))
    vp.config(string("dso:view=adc,trigger=adc<15,timescale=50us"))
    vp.share(0,0)
    Freq.Synth("A",8, 10_000_000)
    repeat
  
```

mark, the code initializes a new frame and processes one video line at a time. For each line, it detects the horizontal sync, skips past the color burst, and then samples the ADC's value every five instructions—for a line length of 240 pixels. To fit a complete video frame into the Propeller's global memory, I store 4 bits of brightness information for each pixel. This data is accessible by all eight cogs on the Propeller. In the DanceBot, one cog is dedicated to run this program continuously to sample video from the camera at 30 fps with a resolution of 240 pixels × 200 lines × 4 bits/pixel.

**Listing 2** is an example program that uses the VideoCapture object. This program configures the clock and imports some objects and then starts the video cog to capture frames. Then, it configures ViewPort to display the streamed video. The Spin code draws a thick black line in the middle of the frame by setting parts of the array to 0. **Photo 3** shows the project's first view of the world.

## REAL-TIME TRACKING

You know how to create the infrastructure to digitize video from a camera into the Propeller's memory using an ADC and one of the Propeller's eight cogs. You have 24 KB of visual data updated 30 times per second. Now you need a filter that can analyze the video and give you just two variables to control the robot.

Start by implementing a filter that identifies the location of the brightest spot in each frame. It's easy to search for the maximum value in your array of pixel brightness values. Just remember

that you're working on 4-bit pixel values compressed into 32-bit longs (A 32-bit long on the Propeller is the basic unit of memory space) (see **Listing 3**).

This filter processes one pixel every five instructions. Because the filter processes only data, not sync marks or color bursts, you can process video at 40 fps. So the filter can easily keep up with the frame grabber object and update the position of the brightest spot in real time.

To integrate this code with the rest of your DanceBot, simply keep this filter code running in its own cog. The cog will continually filter the data provided by the frame grabber and write the x,y location of the brightest spot into the main memory.

You now have two channels of information updated at 30 times/second with which you can drive the two control channels of your robot—speed and direction. Use the x position of the spot to control the turning rate of the robot. If the spot is in the middle, you don't need to do anything. However, if it's on the left side of the image, the algorithm tells the robot to turn left, until the spot is in the center and the robot is facing the source of the spot. A similar technique controls the robot's speed using the vertical position of the spot. The algorithm's goal is to keep the spot's position centered in the image. So, when the spot is too low, the robot is instructed to move forward, which brings the robot closer to the spot's source. Because your camera is looking up at the spot, it will raise the spot in the image. Conversely, if the spot is too high, your robot is too

# Standards Make Sense

Standards improve quality and enable designers to share components across different projects. Today, ARM® Cortex™-M profile processors, combined with the Cortex Microcontroller Software Interface Standard (CMSIS) and optimized middleware from the industry's largest ecosystem, are setting the hardware and software standards for microcontrollers.

These standards enable leading vendors such as Luminary Micro, NXP, and STMicroelectronics to supply advanced microcontrollers, while maximizing code reuse across multiple platforms.

## Cortex-M3 Microcontrollers Make Sense

*"We based our award-winning Stellaris® microcontrollers on Cortex-M3 to provide users with 32-bit performance while eliminating future architectural upgrades or software tool changes."*



**Jean Anne Booth**  
Chief Marketing Officer,  
Luminary Micro



LUMINARY MICRO™

For more information visit  
[www.onARM.com](http://www.onARM.com)

Find us at  
**Embedded Systems  
Conference, San Jose,  
March 30 - April 3rd.**

**ARM - Stand 1502  
Luminary - 1802**

# ARM

The Architecture for the  
Digital World®

© ARM Ltd. AD158-2 | 01.09

close, so it's commanded to drive backwards. Translating this algorithm into code is simple, just scale and offset the x,y location of the spot to control the robot. The complete control program for the robot is posted on the *Circuit Cellar* FTP site.

To illustrate the tracking ability of this filter, I can use ViewPort to display the streamed video with a superimposed trail showing the position returned from the filter over the last minute. Photo 4 shows the grayscale image, as seen by the camera, with a yellow trail showing the path the bright source took.

## LINE FOLLOWING WITH A CAMERA

You can control the behavior of the robot by shining a bright light at the camera. This works in some environments where you can control the lighting and ensure that no other objects reflect or create light to the camera that's brighter than your flashlight. It's also an active method, where you have to power the flashlight. I'll now describe a filter that is less restrictive and uses a passive method to steer the robot.

Most line-following robots use two phototransistors to stay on a line. They're programmed to ensure that one detector is on the dark line while the other is on the lighter background. More sophisticated robots use additional detectors to detect the robot's exact position on the line to look ahead or even to recognize junctions. In this section, you'll build a filter that uses your existing frame grabber to perform line following with a camera.

Again, your frame grabber gives you too much information, so you need to design a filter that will steer a robot in the middle of a line. Tilt the robot's camera so its field of view is from below the horizon to just in front of the robot. Now, you can stream the video to ViewPort and analyze what the video of a properly programmed

**Listing 2**—This Spin program sets the Propeller's clock and includes two objects to find a blob: "Conduit" to stream the video signal to the PC and "VideoCapture" to grab the frame. By continually setting videoFrame[3010] to 0 with the ~ operator, we will have an eight-pixel horizontal black line in the center of the image.

```
CON
    _clkmode      = xtall + pll16x
    _xinfreq      = 5_000_000

OBJ
    vp : "Conduit"          'transfers data to/from PC
    video: "VideoCapture"  'capture video signal pub

findblob|videoFrame[6000],a,blob
vp.register(video.start(@videoFrame,video#HIVIDE0))
vp.config(string("start:video"))
vp.share(@blob,@blob)
repeat
    repeat
        videoFrame[3010]~
```

robot would do. It will become apparent that a good algorithm involves averaging the location of the darkest pixel in each line. This is quite robust, easy to program, and gives a good control signal to the robot. Again, integrating this filter with the rest of the DanceBot is straightforward. Just use the average position of the line to control the direction of the robot while it's moving along at a set speed (see Listing 4).

## TRACK A PATTERN

You've gone from tracking an active, bright spot to following a passive line on an artificial background. Now it's time to track a passive pattern in the real world. The goal for this section is to develop an algorithm and pattern that will steer the robot in any environment. As an experiment, take a

look around and imagine what type of pattern would stand out in our typical cluttered world. For most people, a bar code-like pattern of repeated black and white lines should do relatively well. Of course, this pattern won't suit everyone (e.g., Zebra lovers may need to find another pattern). But this pattern doesn't occur often, and it can be identified reasonably easily with a chain of simple vision filters.

Before you analyze the individual filters, analyze how ViewPort manages a chain of vision filters. Because the Propeller's memory is a limited resource, you can afford only to keep one image in memory at a time. To visualize the effects of different filters, segment the image array into four vision buffers: top left, top right, bottom left, and bottom right. Both the frame grabber and

vision objects support both the full-size and the segmented modes. Filters operate on buffers in that they read data from one place and write their result to another. The DanceBot uses one cog to continuously process image data with a number of filters—one at a time. Configuring the filter's order, parameters, and values is high-level Spin language. Streaming all four vision buffers to ViewPort enables you to watch how each filter manipulates the video in real time.

Now that you



**Photo 3**—This is the system's first picture of a fire truck. Notice the black line at the cross hairs.

**Listing 3**—This Propeller assembly function finds the brightest pixel in the video frame. Thirty-two bits of data are read using the rdlong command. This represents eight pixels, which are inspected one by one by rotating the bits with the ror instruction. The location of the brightest pixel is written to the address pointed to by cmdPtr.

```
doMax
'2 ptrs :src, dnp
'2 value:old, dn
'setup ptrs to positions
    rdlong          val,cmdPtr
    add             cmdPtr,#4
    mov            dnp,src
    add            dnp,bytesNline
    sub            n,#15
    mov            dest,#0 'seexy
    mov            sum,#0 'max value

:loop
    rdlong          old,src
    add            src,#4
    rdlong          dn,dnp
    add            dnp,#4

    mov            m,# 8
    mov            new,#0
    'input: old,dn have pixel in 0..3, new has data
    'output: old,dn rol by 4
:dodiffb
    mov            tmp,old
    and            tmp,#15 'tmp=pixelvalue in mid
    mov            t1,dn
    and            t1,#15 't1=pixel down
    add            tmp,t1 'tmp=two rows
    cmp            tmp,sum wc 'c if tmp<sum
    jmp            #:notMax
    '
    mov            dest,m
    shl            dest,#8
    mov            dest,sum
    shl            dest,#16
    add            dest,n
    mov            sum,tmp 'reset max
:notMax
    ror            dn,#4
    ror            old,#4
    djnz           m,#:dodiffb
    djnz           n,#:loop
    wrlong         dest,val
    jmp            #cmdLoop
```

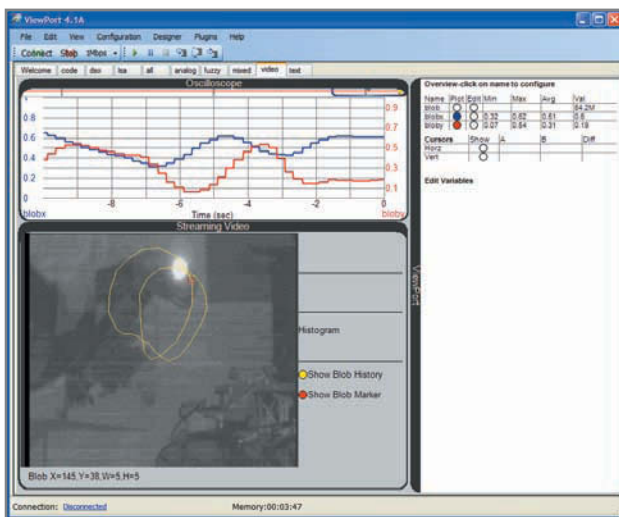
understand how ViewPort vision filters work, build a filter to look for transition edges in your video. Looking at relative changes in value improves the robustness of your algorithm, especially when lighting conditions change. The horizontal Soebel filter is quick to implement and does a good job of detecting vertical edges. To compute

it, replace each pixel with the absolute value of the difference of its horizontal neighbors.

Next, you need to design a filter that will identify regions with multiple strong transitions. The algorithm I chose keeps a running total of the last eight edges by adding the strength of the next edge and subtracting the last.

This running total is saved to the appropriate location in the buffer.

Your last filter finds the location of the maximum in the buffer you just calculated. You can use the same algorithm that you developed to track a bright spot. By chaining these three filters together, you find the location of the maximum running total of vertical transitions. In other words, you can



**Photo 4**—The DanceBot is tracking the path of a flashlight in real time.

# Standards Make Sense

Standards improve quality and enable designers to share components across different projects. Today, ARM® Cortex™-M profile processors, combined with the Cortex Microcontroller Software Interface Standard (CMSIS) and optimized middleware from the industry's largest ecosystem, are setting the hardware and software standards for microcontrollers.

These standards enable leading vendors such as Luminary Micro, NXP, and STMicroelectronics to supply advanced microcontrollers, while maximizing code reuse across multiple platforms.

## Cortex-M3 Microcontrollers Make Sense

*"The strengths of ARM processor-based NXP microcontrollers are fundamentally changing digital products by combining ease-of-use with high connectivity and low power consumption."*



**Geoff Lees**

Vice President and General Manager,  
Microcontroller Product Line



For more information visit  
[www.onARM.com](http://www.onARM.com)

Find us at  
**Embedded Systems  
Conference, San Jose,  
March 30 - April 3rd.**

**ARM - Stand 1502  
NXP - 1010**

# ARM

The Architecture for the  
Digital World®

© ARM Ltd. AD158-2 | 01.09

find a black/white striped pattern.

To make your pattern visible at different distances, repeat the pattern at various scales. When you place this pattern on your belt, you can start dancing with your robot. Stepping closer to the robot makes the image of your pattern move up in the robot's field of vision. This causes the robot to move backwards and maintain a set distance from you. Stepping to one side of the robot causes the pattern to move horizontally, which commands the robot to turn and face you. While dancing with my robot, I discovered some interesting behavior that I hadn't planned on. When I jumped up or crouched down, the robot would change its set distance to me. When I turned around, thereby covering the pattern, the robot also turned around. It no longer detected the pattern and went into search mode where it turned on its axis.

## FIND A BEER BOTTLE

I believe all robot vision articles should include the beer-finding problem. Finding a specially colored beer bottle with a color camera is quite

**Listing 4**—This Propeller assembly function sums the horizontal position of the darkest pixel to steer the robot along a black line. For each horizontal line, it uses t3 to track the location of the minimum brightness and adds this to t2 at the end of the line.

```
doLowest
    rdlong    t1,cmdPtr 't1 is the address of the result variable.
                    We'll write the x position of the line here.
    add      cmdPtr,#4
    mov      t2,#0      't2 is sum of pos

:loopLines    mov      n,linesNpanel 'loop over panel
              mov      x,longsNline 'loop over line
              sub      x,#2
              mov      val,#15      'reset val
:loop         rdlong    old,src      'loop over longpixels
              add      src,#4
              mov      m,#8
              mov      new,#0

:limit        mov      tmp,old
              and      tmp,#15
              cmp      tmp,val wc    'c set if v1<v2
              if_c     add      new,#15
              if_c     mov      val,tmp
              if_c     add      new,#15
              if_c     sub      val,#1
              if_c     mov      t3,x      't3 is pos of lowest item ror new,#4
              ror      old,#4
              djnz     m,#:limit

              wrlong    new,dest
              add      dest,#4
              djnz     x,#:loop      'loop over longpixels
              add      t2,t3
              add      src,#8
              add      dest,#8
              djnz     n,#:loopLines 'loop over lines
              wrlong    t2,t1
              jmp      #cmdLoop
```

doable; however, you're limited to a grayscale camera. You're also not guaranteed that the beer bottle will be the brightest object in the room. There's no line to follow. The beer bottle doesn't have a distinctive pattern. The only available trait is the actual shape of the beer bottle.

Use the correlation algorithm to find the shape of the beer bottle in a typical cluttered environment, u. This algorithm uses brute force to match a desired template to all possible locations in the image. The location of the best match is the location of the beer bottle. The degree of match at any given point is the sum of absolute differences between the pixels of the template and the corresponding pixels of the area to match. To improve the robustness of the algorithm against changes in brightness and contrast, I preprocess the template and every possible match with an auto-level algorithm.

This algorithm does a good job of finding beer bottles. **Photo 5** shows the target identified in a complex environment. However, the image of the bottle must be a close match to the template (i.e., its scale and orientation must be

# GENERAL CIRCUITS CO., LTD

## QUALITY PCB & SERVICE PROTOTYPE TO PRODUCTION

instant online quote  
shopping cart ordering system  
China competitive prices  
free electrically test

web <http://www.pcbcart.com>  
E-mail [sales@pcbcart.com](mailto:sales@pcbcart.com)  
Tel +86-571-87013819  
Fax +86-571-87036705  
Add No.76 GuCui Road, Hangzhou, China

WWW.PCBCART.COM

CHINA PCB SUPPLIER

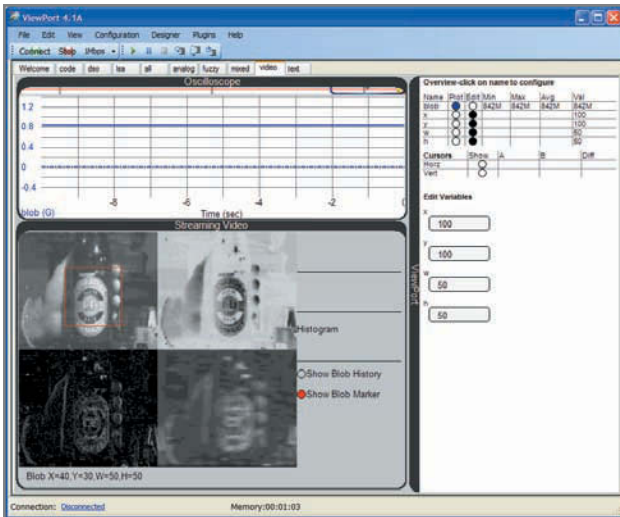


Photo 5—The DanceBot is finding the location of a beer bottle.

identical). With additional processing power, multiple templates could be searched in succession or parallel to more robustly identify the bottle. Once identified, a faster, less processor-taxing algorithm could be used to steer the robot to the beer bottle in real time.

## WRAP UP

I've had a lot of fun building the vision-guided DanceBot with the Parallax Propeller and ViewPort. The Propeller's unique architecture of eight identical cogs made it easy to split my goal of guiding a balancing robot with vision into manageable pieces (see Photo 1b).

*Hanno Sander (hanno@mydancebot.com) has been working with computers since he programmed a lunar lander game for the z80 when he was six. Since then, he graduated from Stanford University with a degree in computer science and then started his corporate career as an Internet entrepreneur. Hanno moved to New Zealand in 2005 to spend time with his growing family and develop sophisticated, yet affordable, robots—starting with the DanceBot. His technical interests include computer vision, embedded systems, industrial control, control theory, parallel computing, and fuzzy logic.*

## PROJECT FILES

To download code, go to [ftp://ftp.circuitcellar.com/pub/Circuit\\_Cellar/2009/224](ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2009/224).

## SOURCES

**C-Cam-2A Miniature video camera**  
Electronics123.com, Inc. | [www.electronics123.com](http://www.electronics123.com)

**ADC08100 ADC**  
National Semiconductor Corp. | [www.national.com](http://www.national.com)

**Propeller**  
Parallax, Inc. | [www.parallax.com](http://www.parallax.com)

Depending on the performance required, I could write the code in the high-level, object-oriented Spin language or dive down to assembly to write completely deterministic code. The logic to configure and control the robot ended up being programmed in Spin, while the frame grabber and vision filters are programmed in assembly. Even though resources are limited, it's possible to carry out advanced

vision processing with it. In today's age of multilevel architectures relying on outside libraries, drivers, and operating systems, it was a breath of fresh air to program the entire chain, from decoding the NTSC waveform to controlling the robot. Visually debugging the DanceBot with ViewPort greatly simplified its development by showing me exactly what was going on with my robot. It acted like a black box when the robot fell down, and showed me what the camera and filters were processing when I was teaching it to dance. It should be straightforward to adapt the code and filters presented in this article to other robots. Good luck! ☑

# Standards Make Sense

Standards improve quality and enable designers to share components across different projects. Today, ARM® Cortex™-M profile processors, combined with the Cortex Microcontroller Software Interface Standard (CMSIS) and optimized middleware from the industry's largest ecosystem, are setting the hardware and software standards for microcontrollers.

These standards enable leading vendors such as Luminary Micro, NXP, and STMicroelectronics to supply advanced microcontrollers, while maximizing code reuse across multiple platforms.

## Cortex-M3 Microcontrollers Make Sense

*"STM32 microcontrollers revolutionize the market by combining high performance and low power with a scalable product range that fits every developer's needs."*



**Daniel Colonna**  
Microcontrollers Division  
Marketing Director



For more information visit  
[www.onARM.com](http://www.onARM.com)

Find us at  
**Embedded Systems  
Conference, San Jose,  
March 30 - April 3rd.**

**ARM - Stand 1502  
ST - Stand 1412**

# ARM

The Architecture for the  
Digital World®

© ARM Ltd. AD158-2 | 01.09



# Web-Enabled Catalog Breaks New Speed Record

## Experience a Faster Way to Search for Components!

Jameco Electronics' new catalog and enhanced Jameco.com website are two tools that are designed to work together to give electronic professionals faster access to the hottest components in the industry.

Color coded references throughout the catalog assist you in analyzing a wide range of brand choices (from franchise sourced to house brands to factory overruns) offering you more pricing options than you'll see anywhere else. Web codes allow you to quickly jump

from catalog to website to view additional specifications, application notes, photos and more products. You'll find exactly what you're looking for without having to wade through hundreds of thousands of products on a complicated website or wielding a twenty pound catalog.

With a flip of the page or a click of the mouse, you have all the tools you need at your fingertips. Reach for Jameco first and order your catalog today.



Order your web-enabled catalog today!

# JAMECO<sup>®</sup>

ELECTRONICS

Call 1-800-831-4242 or Visit [www.Jameco.com/Speed](http://www.Jameco.com/Speed) for the ride of your life!



# Networked Timing

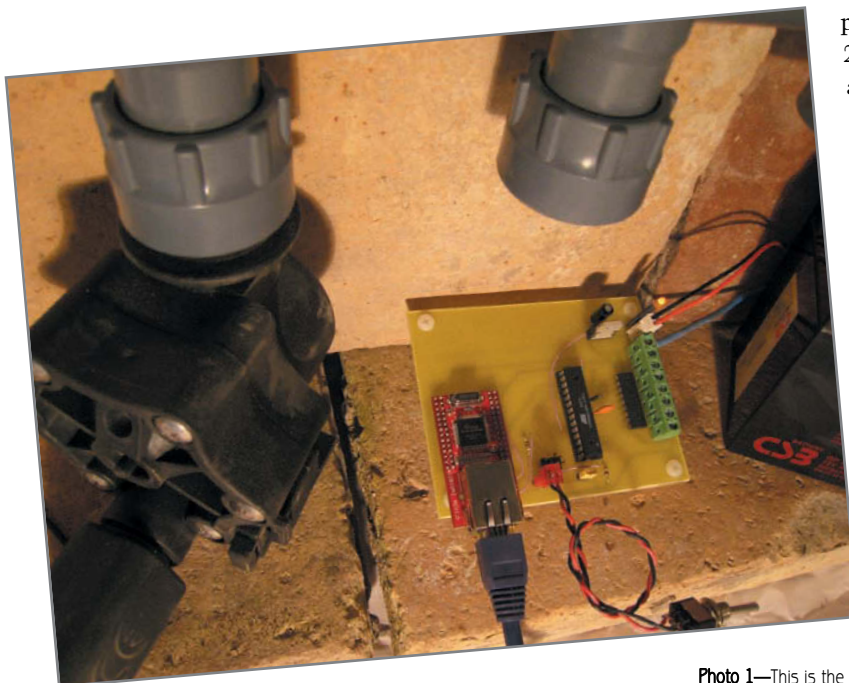
## Build a Timer With Advanced Planning Tools

Precision irrigation control is now a reality. Thomas's irrigation timer with advanced planning (ITAP) is a truly novel irrigation control system. The easy-to-use system, which directs user interaction into a standard web browser, provides useful information such as watering schedules and zone activity.

Electronics are supposed to simplify our lives, but all too often, the reverse is true. While researching irrigation timers, I was struck by how the evolution from an electromechanical design to an electronic design had actually made the device less usable. Early irrigation timers were based on the simple rotary motor design still popular in plug-in appliance timers. A geared motor slowly turned a wheel with on/off pins. If you wanted water at 6 A.M. in zone 2, you simply pushed in the pins for zone 2 and 6 A.M. The wheels and pins also served as crude analog gauges. A glance at the wheel would give you a pretty good idea of when things would next turn on and for how long.

In the electronic age, the wheels and pins are gone. They have been replaced by a tiny LCD and a small keypad. The electronics have added significant new capabilities, but at the cost of requiring significant data entry with a tiny keypad. The visual feedback about what you've programmed is also gone. To ensure that you have not accidentally programmed a 4-h flood, you need to page through identical screens, taking note of each setting.

As you can see in [Photo 1](#), I designed



**Photo 1**—This is the completed circuit board. The WIZnet WIZ810MJ is the red board with the RJ-45 connector. A solenoid valve connected to a manifold is to the left of the circuit board.

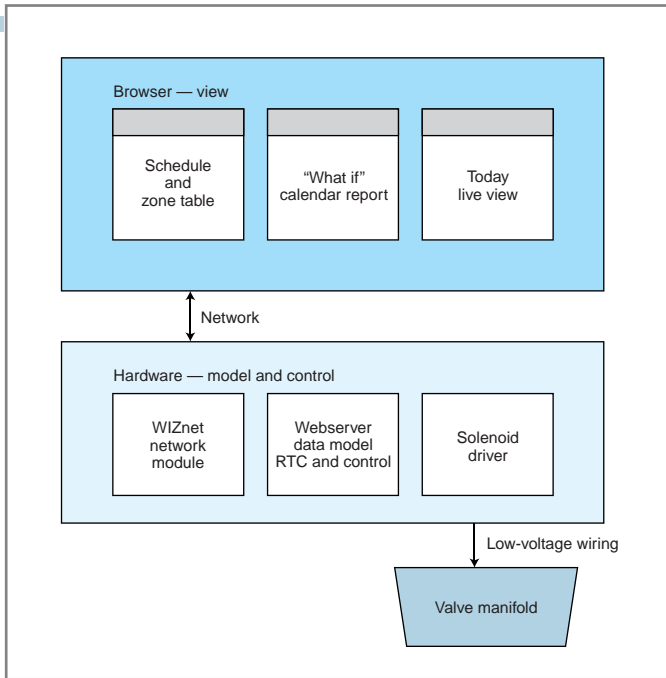


Figure 1—This shows the interaction between the browser pages, the hardware, and the physical zone valves.

electromechanical design. Because the biggest problem was entering and displaying large amounts of data, I selected a web browser for the user interface. I considered various USB-based ways of connecting to a browser, but they invariably required some type of installation on a host PC. Around that time, I saw the WIZnet iEther net Design contest 2007 announcement and decided to build my timer design around the WIZnet WIZ810MJ network module.


A potential design trap in a network-enabled device is the temptation to do too much with the network. An irrigation timer should be a simple device. It is, after all, just a timer and some solenoid valves. The network interface is present in the design because it is the best way to connect to a web browser. I have always been drawn to simple, low-cost designs, so I used the network interface to save money by eliminating an LCD and keyboard. Others might use the network interface as an excuse to jam in seldom-used features and boost the end product's price. Once I started building a web browser interface, I realized that this was a good opportunity to show that web technologies, which are normally associated with large enterprise deployments, can also be used effectively on micro devices.

an Irrigation Timer with Advanced Planning capability (ITAP). The challenge was to incorporate modern functionality, yet keep the design as simple to use as the old

## ITAP DESIGN

The design philosophy was to not consider the ITAP as a networked or web-enabled device, but rather to use the

**Cellular and GPS capable**  
**Data Mover**



- Flash file System
- 4 Chan 12-Bit A/D
- 1MB SRAM
- 512KB FLASH
- 4 Isolated Inputs
- 4 Hi Current Outputs
- 2 External RS-232
- 2 Internal RS-232
- Bat Backed Clk/Cal
- Cell Modem Option
- Internal GPS Option
- Metal Case Option
- 1ma Standby Option

It's easy and cost-effective to do mobile or solar-powered data collection and asset monitoring with the JK micro's **Data Mover**. With the ability to integrate a Cellular Modem, GPS and DOS-based embedded controller in a single rugged enclosure, you can capture and transmit your data quickly, easily and at low cost. Inexpensive development kits including Borland C/C++ and PowerBasic are available now. Call or email us for more details.

Call **530-297-6073** Email [sales@jkmicro.com](mailto:sales@jkmicro.com)  
 On the web at [www.jkmicro.com](http://www.jkmicro.com)

**JK microsystems**



**AP CIRCUITS**  
 PCB Fabrication Since 1984

---

As low as...  
**\$9.95**  
 each!

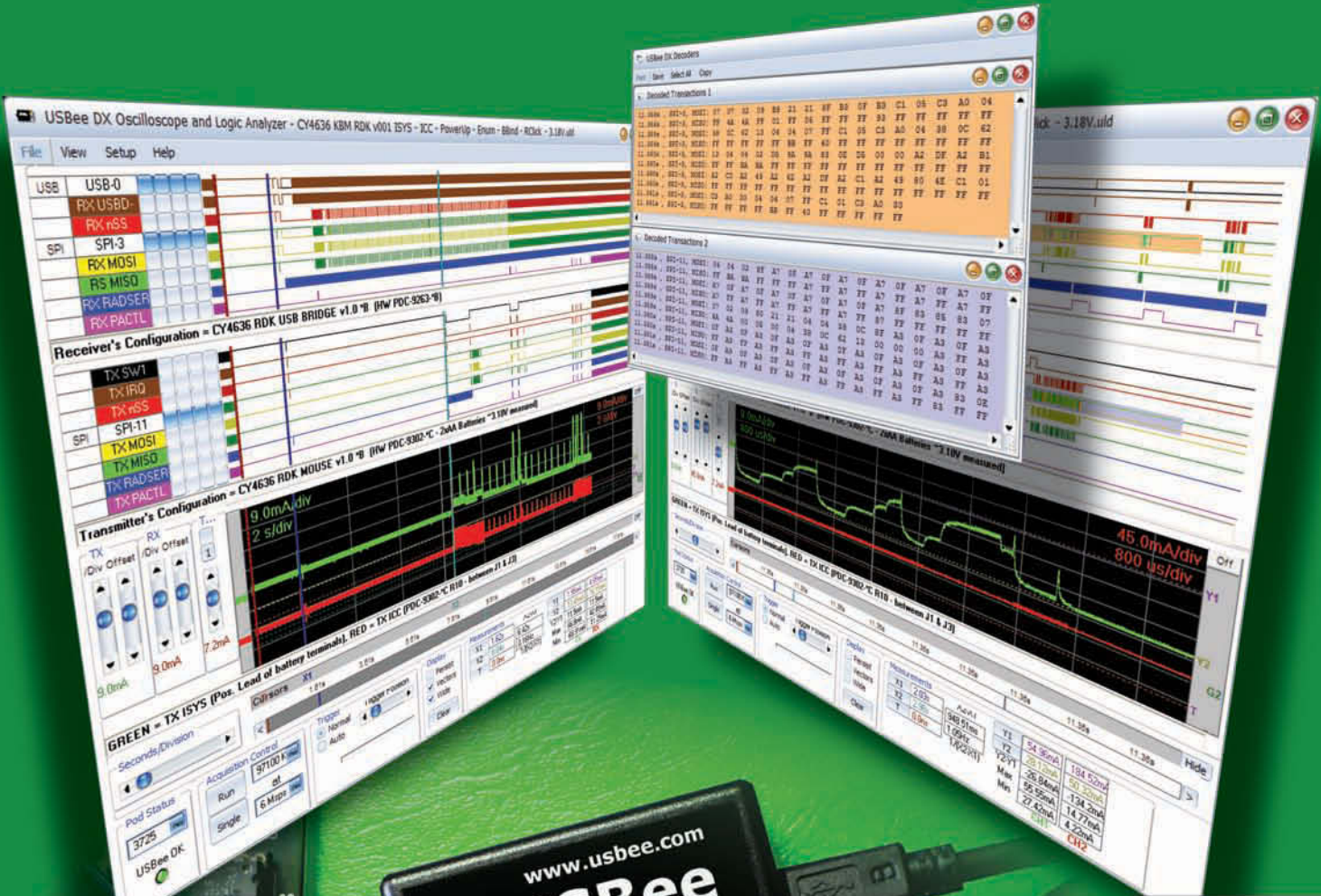
Two Boards  
 Two Layers  
 Two Masks  
 One Legend

**Unmasked boards ship next day!**

**[www.apcircuits.com](http://www.apcircuits.com)**

---





Actual Size

| USBee ZX | USBee AX - Standard | USBee AX - Plus | USBee AX - Pro | USBee DX |
|----------|---------------------|-----------------|----------------|----------|
|----------|---------------------|-----------------|----------------|----------|

|                                  | USBee ZX | USBee AX - Standard | USBee AX - Plus | USBee AX - Pro | USBee DX |
|----------------------------------|----------|---------------------|-----------------|----------------|----------|
| Oscilloscope                     | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Logic Analyzer                   | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Mixed Signal Oscilloscope        | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Digital Signal Generator         | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Digital Voltmeter                | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Bus Data Extractors              | ✓        | ✓                   | ✓               | ✓              | ✓        |
| USB (Low and Full Speed) Decoder | ✓        | ✓                   | ✓               | ✓              | ✓        |
| I2C Decoder                      | ✓        | ✓                   | ✓               | ✓              | ✓        |
| SPI Decoder                      | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Async Decoder                    | ✓        | ✓                   | ✓               | ✓              | ✓        |
| CAN Decoder                      | ✓        | ✓                   | ✓               | ✓              | ✓        |
| I2S Decoder                      | ✓        | ✓                   | ✓               | ✓              | ✓        |
| 1-Wire Decoder                   | ✓        | ✓                   | ✓               | ✓              | ✓        |
| SM Bus Decoder                   | ✓        | ✓                   | ✓               | ✓              | ✓        |
| PS/2 Decoder                     | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Parallel Decoder                 | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Serial Decoder                   | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Click and Drag Bus Decoding      | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Data Logger                      | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Frequency Counter                | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Remote Controller                | ✓        | ✓                   | ✓               | ✓              | ✓        |
| PWM Controller                   | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Frequency Generator              | ✓        | ✓                   | ✓               | ✓              | ✓        |
| I2C Controller                   | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Pulse Counter                    | ✓        | ✓                   | ✓               | ✓              | ✓        |
| USBee Toolbuilder Source Code    | ✓        | ✓                   | ✓               | ✓              | ✓        |
| Digital Channels                 | 8        | 8                   | 8               | 8              | 16       |
| Maximum Sample Rate (MSPS)       |          |                     | 24              |                |          |
| Buffer Depth                     |          |                     |                 |                |          |
| Analog Channels                  | 0        | 1                   | 1               | 1              | 2        |
| Analog Input Range               |          |                     |                 |                | +/-10V   |

Powerful Debugging - Small and Portable

[www.USBee.com](http://www.USBee.com)

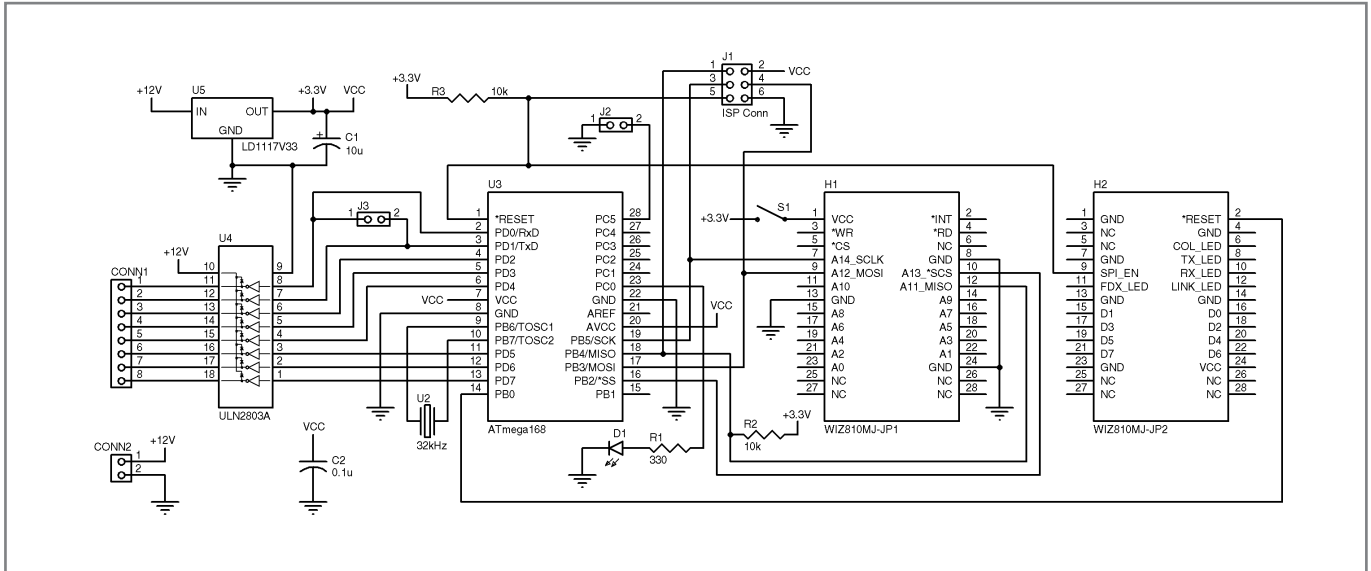


Figure 2—Connectors H1 and H2 are for the WIZ810MJ module. In SPI mode, few of the WIZ810MJ's pins need to be connected.

network as a better serial line. Conceptually, the ITAP has two boxes. Box one is a headless control unit that mounts in the garden and is hard-wired to zone solenoids. It also has an internal calendar and timer that run the watering schedule. Box

two is a fancy LCD and keyboard that plugs into box one whenever a change of schedule is required. Box two, of course, is really a laptop with a web browser.

The hardware box represents the headless control unit (see Figure 1).

A microcontroller maintains a data model in EEPROM that is used by the timer-control firmware to determine when to turn on and turn off solenoid water valves. A WIZ810MJ network module is used by the web server firmware to accept requests

## Easy Embedded Linux

**\$169**  
Qty 1

**16MB FLASH / 32MB RAM**  
**200Mhz Arm9 CPU**  
**16 Digital I/O**  
**Watchdog**  
**10/100 Ethernet**  
**Battery backed Clock/Calendar**

**Audio In/Out**  
**2 USB**  
**2 Serial Ports**

*We brought you the world's easiest to use DOS controllers and now we've done it again with Linux. The **OmniFlash** controller comes preloaded with Linux and our development kit includes all the tools you need to get your project up and running fast.*

*Out-of-the-box kernel support for USB mass storage and 802.11b wireless, along with a fully integrated Clock/Calendar puts the **OmniFlash** ahead of the competition.*

Call **530-297-6073** Email [sales@jkmicro.com](mailto:sales@jkmicro.com)  
On the web at [www.jkmicro.com](http://www.jkmicro.com)

# JK microsystems



## Embedded & Network Computing Technologies

### Tiny, Light & Powerful All In One!



**Embedded Computer**  
Tynycore form-factor (36 x 41 mm)

**ATMEL AT91SAM9G20 @ 400MHz**

256MB NAND Flash (8bits),  
64MB SDRAM (32bits @ 133 MHz)  
USB Device, Serial DBGU & 2 Expansion Ports.

[www.calao-systems.com](http://www.calao-systems.com)

**Listing 1**—This shell script compresses the HTML and JavaScript files as part of building the firmware image. The compressed files reside in the Atmel ATmega168's flash memory.

```
#!/bin/sh
FILES="error.html index.html edit.html common.js cal.html
day.html"
(
for f in $FILES; do
    id=`echo $f | tr '.' '_'`
    echo "prog_uchar $id[] = {"
    gzip -9 -v $f | xxd -i
    echo "};"
done
) > gz_data.inc

SCK1
__builtin_write_OSCCONL(0x40); //lock
```

from a standard web browser. The browser pages provide data entry and display capabilities of the watering schedule, as well as calendar-based planning tools. The entire design is self-contained. There are no extra files, scripts, or drivers that need to be installed on a PC.

An irrigation timer is not something you reprogram often. After an initial period of fine-tuning, the unit is expected to work, unattended, for the rest of the season. I did not need to add any daily reporting, only a minimal status page. This emphasizes the design point that the network connection is only there for reprogramming the device. Fancy reports and real-time status pages are engaging for the first few weeks of ownership, but then for most people the excitement of watching the grass grow begins to fade.

The long periods of time between reprogramming make it so you don't have to install any information on a PC. It is usually not a problem to pop in an installation CD when you first buy a product. Assuming there are no missing drivers or system conflicts, the first install is easy. The problem comes six months later, after you have upgraded your PC and cannot find the installation CD. For the same reason, it is important that all HTML pages live on the device

and not on a PC. If the device requires a PC installation, it stops being a stand-alone device and becomes yet another peripheral ready to break when you modify your PC.

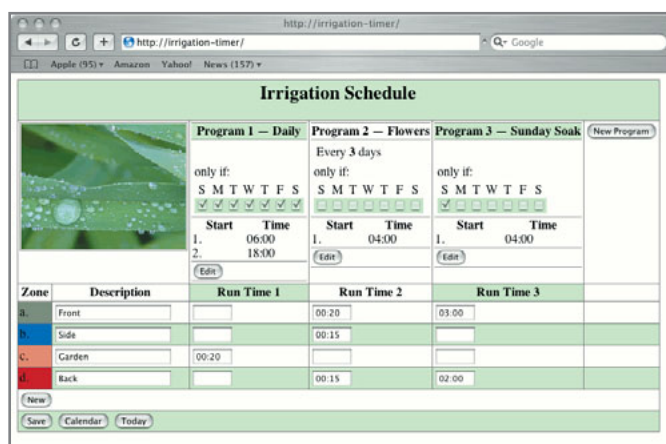
## HARDWARE

The hardware was designed to include the fewest number of parts. The essential parts are an Atmel ATmega168, a WIZ810MJ network module, and a ULN2803 Darlington array (see [Figure 2](#)). I do all of my development work in Linux. The excellent AVR toolchain available for Linux is one of the reasons I prefer Atmel microcontrollers. With Linux command line tools, I can do all of my firmware development at my desk, not hunched over the lab bench. An old laptop sits on the lab bench and serves as a network-to-USB gateway for the in-circuit programmer. I chose

the ATmega168 largely because of its 16 KB of flash memory and because it is easily available in a DIP package, simplifying prototyping. Timer 2 is used with an external 32-kHz crystal in real-time clock mode. The SPI master controls the WIZ810MJ. Other than a handful of I/O lines, none of the microcontroller's other features are used. Although it seems like a shame to use so few features, it would be pointless to complicate the design.

The WIZ810MJ is a network module that takes care of most of the complexity of adding TCP/IP networking to a design. In SPI mode, only a handful of the 56 pins are needed. The rest can be left as no connects. The version of the module that I worked with has a known problem in that it continues to drive the SPI lines even when its \*SS is unasserted. The work-around is to drive the module pin SPI\_EN low, which will free up the other SPI lines. The ATmega168 uses the SPI lines for serial programming, so I had to do the SPI\_EN trick even though there are no other SPI devices. The WIZ810MJ draws a fair amount of current, so I did not want to leave it powered up continuously. I toyed with the idea of adding circuitry to power the module on and off. Because the network link LED status is available on a module pin, I could periodically power on the module, check for an active link, and turn it off. In the end, I just put in a toggle switch, which was crude but effective.

A ULN2803 is used as the solenoid driver. It is an eight-driver package with internal clamping diodes. The inputs are logic-level and were connected directly to port pins on the microcontroller. Each driver can sink 500 mA, which is ample for a typical irrigation solenoid valve drawing about 200 mA at 12 VDC. The solenoids are 24 VAC, but are quite content running on 12 VDC. Conveniently, inputs and outputs



**Photo 2**—The irrigation schedule is the main browser page. Four solenoid valves are represented by zones a-d. The three different programs determine at which time and on which days the zones will be active.

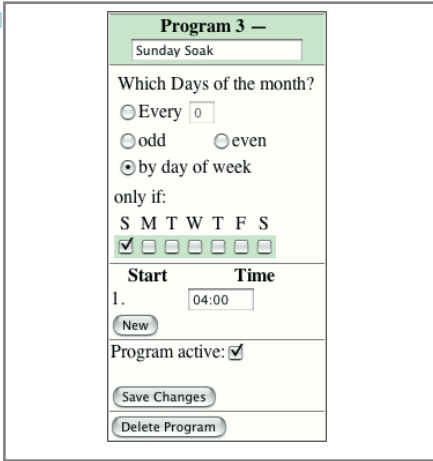


Photo 3—The program details on the main page are read-only. Clicking “edit” brings up an editable version.

are on opposite sides of the DIP package, making layout much easier on a single-sided board.

The only user feedback from the hardware are the WIZ810MJ’s two network status LEDs and an LED driven by the microcontroller. The microcontroller-driven LED blinks out either a hint at the IP address or an indication of solenoid status.

## DATA MODEL

There are a number of ways to organize a watering schedule. Several vendors have agreed on a single model, which is the one used here. First, there are zones, which correspond to a physical run of pipe controlled by a solenoid valve. Then there are programs, which indicate which days of the week and at what time of the day to start an action. Make a table and put the zones in the rows and the programs in the columns. At the intersections, decide if and for how long a zone should be on for each program. This can get somewhat complex, but it provides a good deal of flexibility for things like long, slow soakings and for following community odd/even watering restrictions.

The ITAP’s internal data model consists of an array of program structures that keep all of the specifics for a

program, including start times and run length times, for each zone. All times are kept in minutes rather than hours and minutes. Internally, this simplifies storage and comparisons. A running count is kept of the day of the year in addition to the day of the month. The former is used for computing every N day cycles. The latter is used for odd/even days of the month.

## SOFTWARE

The software divides into microcontroller support, network code, data model access, file storage, and timer logic. The microcontroller support code is minimal. The only interrupt is the Timer 2 overflow, which is configured as a 4-Hz RTC. Everything else is done by polled I/O. Every couple of seconds the main loop checks if the WIZ810MJ has been powered up. A simple read of the last byte of the IP address is performed. If the module is off, the byte will read as 0xFF. If the byte matches the expected value stored in EEPROM, the WIZnet module is assumed to be present and it is reinitialized.

The ATmega168’s PORT D connects pin for pin to the ULN2803 solenoid controller. Thus, pin zero of PORT D corresponds to solenoid zero, pin one to solenoid one, and so on. At each program step, there is the possibility that you might request all zones to switch on. To

limit current inrush problems, the software sets the values bit by bit with a small delay in between.

In a similar software rather than hardware role, the WIZ810MJ’s reset line is controlled by a port pin rather than RC logic. In addition to parts savings, this allows for a full part reset.

The network code is designed for the specific task of communicating with the one browser that it expects to find on a private network. The main loop polls the WIZnet sockets for a page request. Only HTTP GET requests are honored. The request can be for either a file stored in the flash memory file system or for a data page built on the fly. Changes to the data model are accomplished with arguments of the page request.

In a system where a browser request causes some action, such as writing EEPROM or switching on a sprinkler, minimize the chance that a user unintentionally repeats the action by clicking the REFRESH button. The method used here is to clean up the location line by returning a 307 REDIRECT result code. With no arguments to the page request, pressing REFRESH will simply reload the page with no additional actions.

When using the WIZ810MJ as a web server, there is a potential trap if the pages being served are complex. The W5100 can support a maximum of four simultaneous connections.

If the HTML page rendered by the browser has more than four subelements (e.g., <image> or <script> tags), the browser will likely issue simultaneous requests. If a WIZnet socket is in LISTEN state, it will accept a connection. If the browser attempts a connection for which there is no listening socket, the browser will get a connection-refused error. Depending on the browser, this could result in either a reported error or a noticeable delay before a retry. The workaround is to keep the pages simple, with few

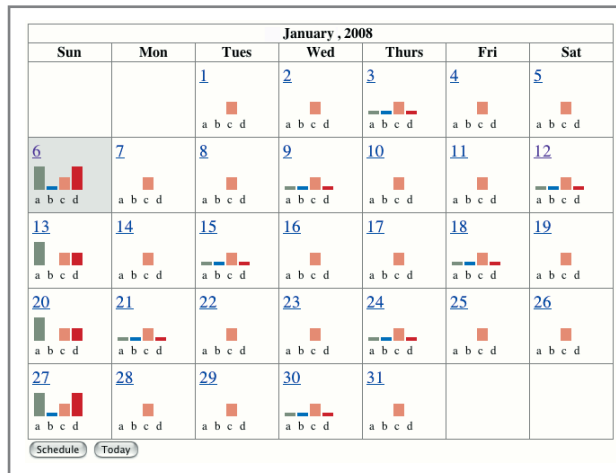


Photo 4—The calendar view shows what will happen this month. The bars in the bar graph correspond to the zones. The height of the bar shows the relative time that the zone will be on that day.

subelements. Newer chips, such as the WIZnet W5300, support up to eight sockets.

An early design trade-off in a project like this is deciding how much flash memory to devote to storing HTML page data. It would have been easier in many ways to just wire in a 1-GB SD flash memory card and fill it with the output from a web site builder tool. However, this would have negated the strong desire I had to show by example that an attractive, useful, and feature-complete UI could be implemented on a small-footprint device.

Accepting the limitations of finite storage, the question was how to stay as small as possible. Early in the design process, I considered keeping HTML templates in some sort of compressed form. The plan was to decompress the templates, fill in the actual values, and then serve this modified data to the browser. The final design is better in every way.

If the pages are not modified but instead have static content, they can be highly precompressed with the gzip utility and simply copied, unmodified, from flash memory to the network. No compression code is needed. The pages are just a payload. Modern browsers already accept gzipped data, so nothing additional is required.

Using JavaScript makes static content possible. The JavaScript language has constructs for dynamically building pretty much anything that can be done with HTML. All of the ITAP tables and forms are dynamically built. The data needed to fill in the tables and forms is formatted using JavaScript Object Notation (JSON) and resides in a separately loadable JavaScript file. The file is

built on the fly each time a server request is made for "pdata.js." In a larger system, data would likely be passed as XML. XML is a bulky format intended for data exchange between unrelated systems. XML really has no place in point-to-point micro applications. The nice thing about using JSON is that it is automatically parsed as it is read by the browser. Additional parsing is not required.

Another important benefit associated with putting all of the UI building code in JavaScript is that there is a clean separation between the UI view and the data model. The data model is maintained by the ITAP firmware. The firmware knows nothing of UI layout. A change to the UI does not require a firmware change.

## COMPRESSED FILES

Looking through the WIZnet sample code, I noticed copyright notices around the code that accesses flash memory-based files. Thus, I thought it would be useful to show my standard tools approach to compiling files into flash memory (see Listing 1). This Linux shell script takes each file, compresses it, converts it to ASCII hexadecimal, and wraps the result in a data declaration ready for a C language `#include` statement.

## USER INTERFACE

Photo 2 shows the ITAP's main schedule page. To keep the size small, only standard buttons and standard fonts were used. The schedule page is implemented as one large HTML FORM. Inside the FORM is a TABLE built dynamically with JavaScript. Only a small bit of HTML is used to define the basic




**MACH64**  
PROGRAMMABLE LOGIC  
STARTER KIT

Based on the Lattice ispMach 4064.

Includes 250 page lab manual.

Learn CPLDs the fun way with the MACH64! This complete kit comes with everything you need to take you from mystery to mastery with CPLDs and programmable logic. Learn to turn *software* into *hardware*!

www.XGAMESTATION.COM




**PIC-SERVO**  
MOTION CONTROL

MOTION CONTROLLERS FOR BRUSH, BRUSHLESS AND STEPPER MOTORS.

- controller chips
- controller boards

www.picservo.com  
JEFFREY KERR, LLC



**THE SERIAL PORT LIVES!**

Everything you need to know about COM ports, USB virtual COM ports, & asynchronous serial ports for embedded systems.

Hardware & software for RS-232 & RS-485. Wireless options and more.

**Serial Port Complete**  
Second Edition  
Jan Axelson

ISBN 978-1-931448-06-2 \$39.95  
Lakeview Research LLC www.Lvr.com

**From the author of USB Complete**

prev **TODAY — January 6, 2008 6:06** next

| Zone | Description | Set | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 |
|------|-------------|-----|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| a.   | Front       | On  |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| b.   | Side        | off |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| c.   | Garden      | On  |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| d.   | Back        | off |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |

Schedule Calendar Today

Photo 5—Clicking on a specific day in the calendar view brings up this page, which shows when the zone will be active.

table and headings. Each page in the UI contains the same first line:

```
<script language="JavaScript"
  type="text/javascript"
  src="pdata.js"></script>
```

This code loads the current program and zone data. Based on the number of programs and zones, columns and rows are dynamically added and populated with their values. JavaScript functions take care of details like converting between minutes and hour and minute values. There are three programs and four zones in Photo 2. The empty fields indicate that the corresponding zone is off during that program.

To keep the main schedule page from getting too cluttered, only a summary of the program values is shown, and they are all read-only. Clicking an Edit button brings up a program edit page (see Photo 3). Both program tables are generated by the same JavaScript code. A flag tells the JavaScript to hide or desensitize certain fields in read-only mode.

The basic program interval can be set to either a fixed number of days or odd/even days. The check box by day of week is just an aid. It is no different than selecting a one-day interval. The days of the week check boxes are used to modify the basic interval. In an earlier design, I had two rows of check boxes, one for days when the program should run and a second for days when the program should not run. This was not necessary because a "not Friday" program is logically the same as selecting every day except Friday. There is always a trade-off between convenience and UI clutter. The Program active check box is used with the planning tools. To see what a particular program contributes to the overall totals, it can be temporarily disabled.

As you can see, there is plenty of flexibility, but also plenty of opportunity to make a mistake. The planning tools are a simple, but effective innovation that gives a glimpse into the future. Based on the currently active programs, the firmware runs

the clock forward to determine how much water each zone will get each day. This summary data is used to construct the calendar page (see Photo 4). Each day in the calendar contains a small bar chart. The letters and colors of each column correspond to a zone from the main schedule page. For simplicity, the height of the bar is limited to one of four discrete steps. The bar graph is not a GIF or JPEG image. Instead, it is implemented in JavaScript as a four-row table with variable ROWSPAN elements. As with the rest of the UI, the firmware reports only the data. The JavaScript makes all display and layout decisions.

The final page is the Day page, which drills down from a calendar bar chart to show when (during the day) the water will run (see Photo 5). Each tick mark corresponds to a 15-min. interval. The tick marks are constructed from TABLE rows with a small white border to highlight the individual tick. Together the Day and Calendar pages make for accurate "what if" planning.

I did not originally plan to have status view or zone on/off override buttons because I did not expect to have the network connected except during reprogramming. I later added these features mainly for debugging and

demonstration purposes. The status and override are implemented with the JavaScript XMLHttpRequest() facility, the cornerstone of AJAX.

## IMPROVEMENTS

The ITAP was a fun project to build. It was much more of a software project than a hardware project. The design has proven to be simple to use and easy to explain to others. There is still a learning curve to understand what zones and programs are all about, but this knowledge is also required for any timer. The fact that there is no expensive LCD sitting idle in the garden is continuously comforting.

One serious drawback with the current design is having a static IP address for the ITAP. I did not want the ITAP to get its address by DHCP, because I expect to have a laptop plugged into the ITAP while standing in the garden. A reasonable alternative would be for the ITAP to play DHCP server for a laptop client.

WIZnet has released two additional modules, the WIZ830MJ and WIZ812MJ. I have not worked with either, but both appear to be better choices for new designs. The WIZ830MJ has the W5300 chip. The WIZ812MJ is a redesign of the WIZ810MJ. Both new modules have 2.54-mm headers. ☐

*Editor's note: This project won First Place in the 2007 WIZnet iEther net Design contest. For more information about this design and the other winning projects, go to [www.circuitcellar.com/wiznet](http://www.circuitcellar.com/wiznet).*

*Thomas Bereiter (itimer@micaview.com) has written software for everything from microcontrollers to huge distributed systems. He has a B.S. in computer science from the University of Illinois. Thomas currently designs prototype systems in Umbria, Italy.*

## PROJECT FILES

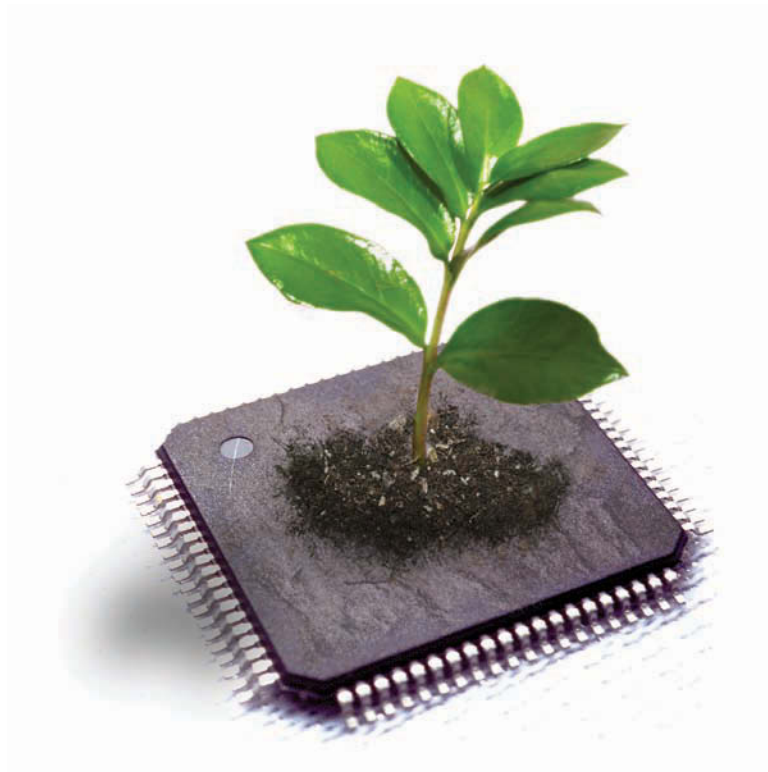
To download code, go to [ftp://ftp.circuitcellar.com/pub/Circuit\\_Cellar/2009/224](ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2009/224).

## SOURCES

**ATmega168 Microcontroller**  
Atmel Corp. | [www.atmel.com](http://www.atmel.com)

**WIZ810MJ Network module and W5100/5300 Ether net controller**  
WIZnet, Inc. | [www.wiznet.co.kr/en/](http://www.wiznet.co.kr/en/)





With compilers that cut power consumption

# Your MCU can go green!

**HI-TECH C<sup>®</sup> PRO**  
ANSI C Compilers

featuring Omniscent Code Generation™ (OCG)

OCG is a new whole-program compilation technology that cuts power consumption by reducing interrupt latency and increasing speed while simultaneously reducing code size. It's a win-win for your project and the environment.

HI-TECH C PRO compilers: available for 8051, Cypress<sup>®</sup> PSoC<sup>®</sup> Programmable System-on-Chip, and Microchip PIC10/12/16/18/32 MCUs.

Denser code, better performance: [www.htsoft.com/ocg](http://www.htsoft.com/ocg)

**HI-TECH**  
S O F T W A R E

**OCG**

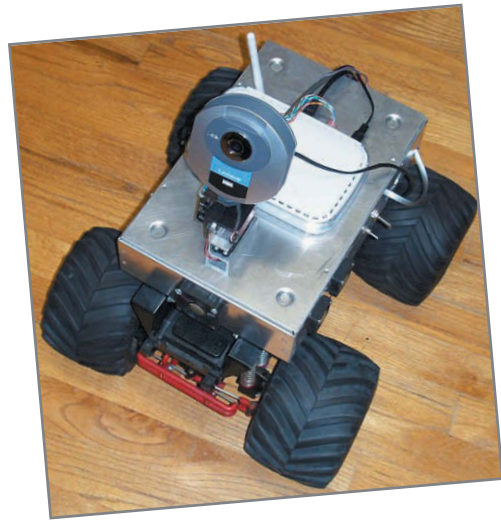
[www.htsoft.com](http://www.htsoft.com) • [sales@htsoft.com](mailto:sales@htsoft.com) • +1 800 735 5715

Cypress and PSoC are registered trademarks of Cypress Semiconductor Corp. All other trademarks are the property of their respective owners.

# Wireless Mobile Robotics

## A Wi-Fi-Enabled System With a Mounted Webcam

Scott used a microcontroller, an embedded Ethernet board, and a wireless router in an innovative control system for a compact mobile robot. The robot features a mounted webcam that transmits real-time pictures to a remote laptop. Scott explains how he planned the project, assembled the pieces, and created the control software.



**Photo 1**—This is the assembled WiFi-PIC-Bot with a webcam. Pan and tilt servos are located just below the camera.

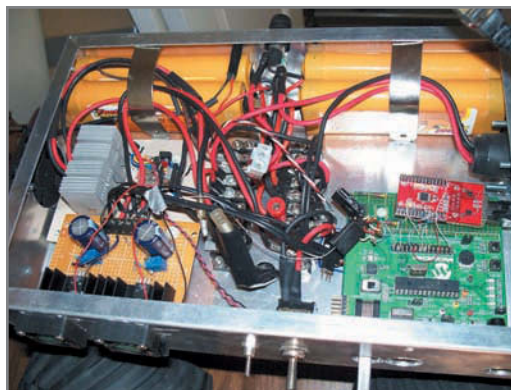
Robots are everywhere. They are used to build cars. They are sent to poke at rocks on Mars. Small rovers can vacuum your pool or your house, and there are several different models at the local toy store. I've played with RC cars, boats, planes, and helicopters for many years, but after seeing TV shows like *BattleBots* and *Junkyard Wars*, I had an itch to build something bigger and better to play with.

In 2001, I volunteered as a mentor for a FIRST Robotics Competition at Penn High School in Mishawaka, IN. It was an incredible experience. (If you like building mechanical devices, I recommend that you consider becoming a mentor.) The FIRST robots use controllers similar to standard RC types, which are capable of telemetry and feedback, but they are a bit pricey for the average designer. These days, however, you can pick up a Wi-Fi router for approximately \$30 and a used laptop for as little as \$100. The two of these, along with a fast Microchip Technology PIC microcontroller and a WIZnet embedded Ethernet board, make a nice platform for a robot project.

In this article, I will describe how I built a robotics system—which I call the “WiFi-PIC-Bot”—along with the control software and interface electronics (see [Photo 1](#)). I entered this project in the WIZnet iEthernet Design Contest 2007.

But since then, the robot has gone through a few upgrades. You can view the project as it was then ([www.circuitcellar.com/wiznet/winners/DE/001106.html](http://www.circuitcellar.com/wiznet/winners/DE/001106.html)).

The WiFi-PIC-Bot is a modified RC ClodBuster (dual-motor, four-wheel drive with four-wheel steering) monster truck. I replaced the RC servo receiver with a WIZnet WIZ810MJ embedded Ethernet board controlled by a Microchip Technology PIC24FJ64GA002. The WIZnet board is connected to a Netgear WGR614 wireless router that transfers steering and throttle servo data as UDP packets back and forth to a remote laptop. The PC program reads a joystick, sends the servo commands out through its Wi-Fi card, and displays a real-time picture from a Linksys WVC54GC Internet-ready webcam mounted on the robot. One of the latest upgrades was pan and tilt servos for the Linksys camera.



**Photo 2**—In the control box, battery packs are at the top, power supplies are at the bottom (left), and speed control is in the middle (left). A Microchip DM300027 development board with a WIZnet WIZ810MJ Ethernet board is in the lower right.

### MODIFICATIONS

Making an RC monster truck into a robot isn't a simple task. Each issue had to be addressed separately. Many of the original design components had to be modified to

# Sweet!

## Introducing the MiniCore™ Series of Networking Modules

Smaller than a sugar packet, the Rabbit® MiniCore series of easy-to-use, ultra-compact, and low-cost networking modules come in several pin-compatible flavors. Optimized for real-time control, communications and networking applications such as energy management and intelligent building automation, MiniCore will surely add sweetness to your design.

- **Wireless and wired interfaces**
- **Ultra-compact form factor**
- **Low-profile for design flexibility**
- **Priced for volume applications**

Wi-Fi and  
Ethernet  
Versions



### MiniCore Module Development Kits

From  
**\$49** Limited  
time offer.

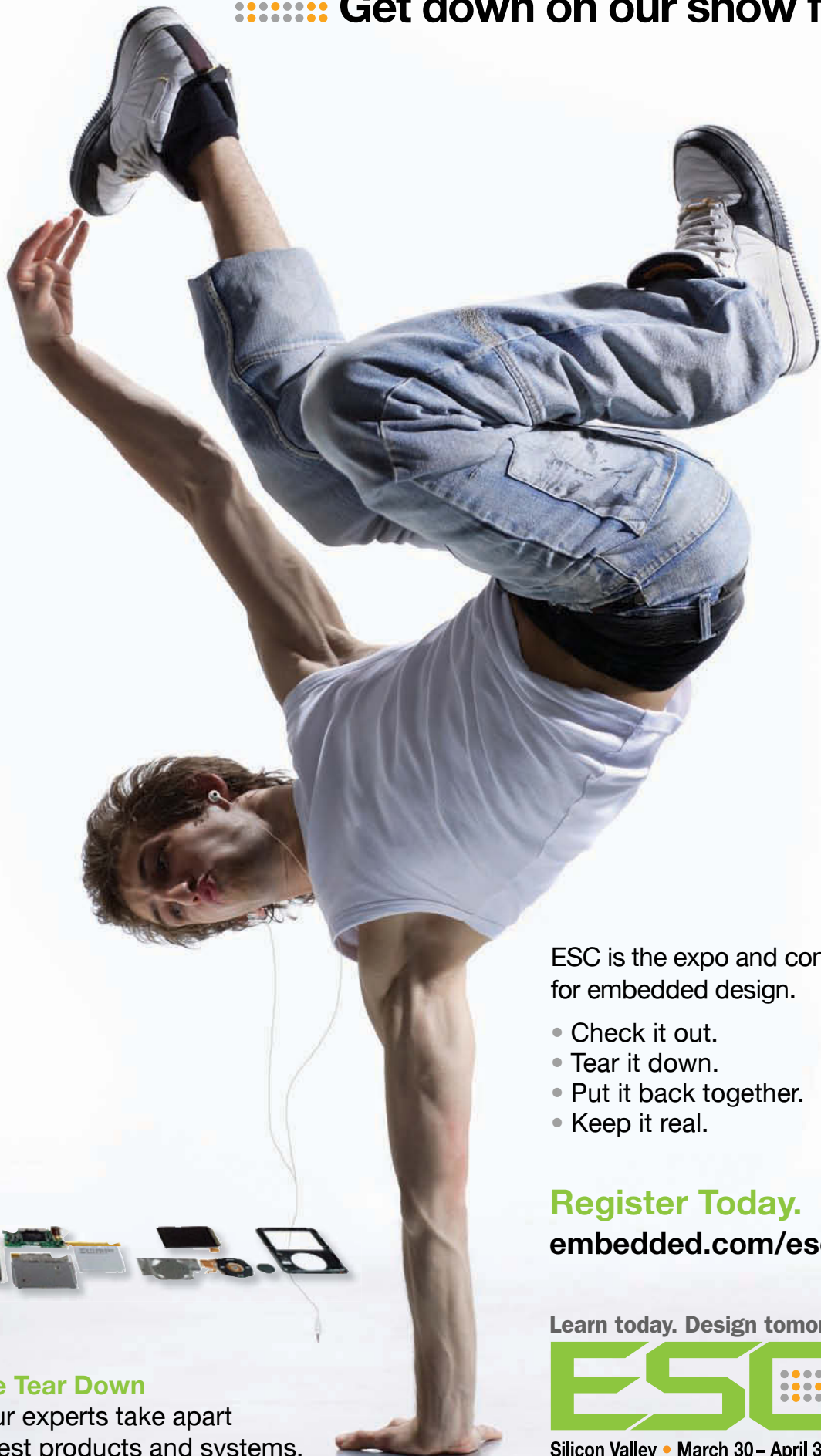


Buy now at: **1.888.411.7228** • [rabbitwirelesskits.com](http://rabbitwirelesskits.com)

1.888.411.7228  
[rabbitwirelesskits.com](http://rabbitwirelesskits.com)  
2900 Spafford Street, Davis, CA 95618



 **Get down on our show floor.**



ESC is the expo and conference for embedded design.

- Check it out.
- Tear it down.
- Put it back together.
- Keep it real.

**Register Today.**

[embedded.com/esc/sv](http://embedded.com/esc/sv)

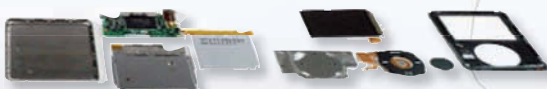
Learn today. Design tomorrow.



Silicon Valley • March 30 – April 3, 2009

**Do the Tear Down**

See our experts take apart the latest products and systems.



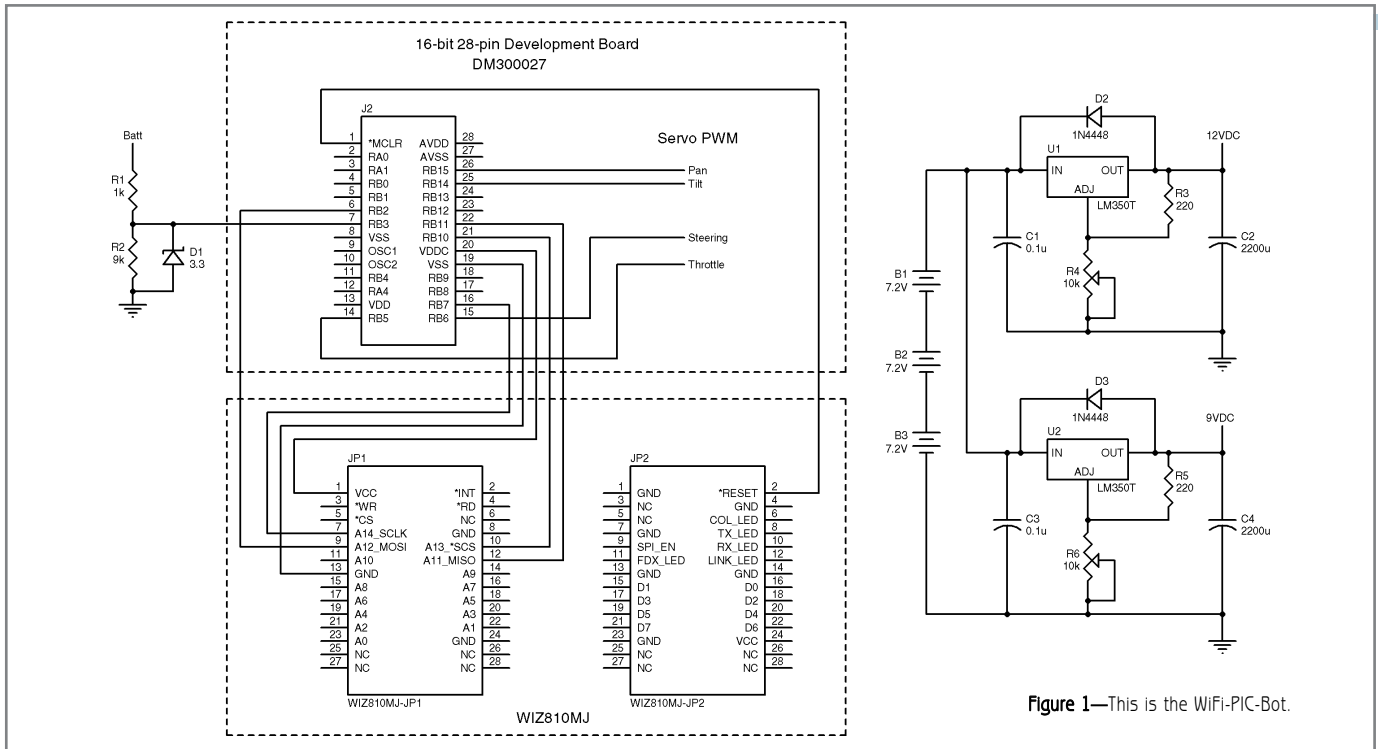


Figure 1—This is the WiFi-PIC-Bot.

make the toy into a reliable and sturdy robot platform. The ClodBuster's single 7.2-V battery wasn't sufficient to run the vehicle for more than a few minutes, and I couldn't use it for running the new processor, adding servos, the webcam, and the wireless router. The stock body and suspension system couldn't carry much weight. The original motor gears were fine for zipping around the yard, but were a bit too fast for indoor use. The stock motor control was just a four-position mechanical switch operated by a small servo. Slow speed was accomplished by switching a resistor in series with the motors. Each of these issues had to be addressed separately.

The first parts to go were the wimpy plastic body, the RC receiver, the mechanical speed control, and the stock battery packs. An aluminum 8" x 12" x 2.5" box fit nicely on top of

the frame. It had enough room inside for most of the electronics and batteries. I mounted power supply and speed control cooling fans, power LEDs, and switches to the outside of the box. I drilled for servo wire access, a programming connector, and network cable ports. I added a multi-terminal, high-amperage connector for on-system battery charging with bank selection. I attached the Internet camera with its pan and tilt servos along with the Wi-Fi router to the top of the box.

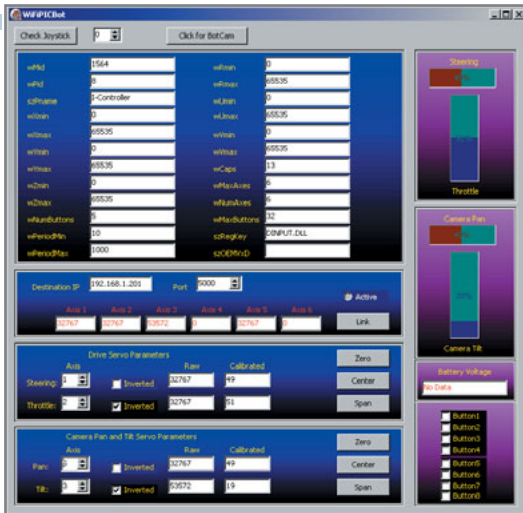
Both stock drive motors were already changed to high-performance models, so I left them as they were. I changed the transmission gears to the highest gear ratio I could find to slow down the robot for indoor use and to add torque for moving the extra weight around. I added hard rubber supports inside the springs to the suspension system shocks to accommodate the added weight of all the batteries and new electronics.

The main motor drive speed control had an interesting set of issues to resolve. Two parallel high-performance motors operating at high torque require a high-current controller with reverse, and I didn't have one. However, I had a Novak T-4 electronic speed control in my stash of RC parts and a collection of high-current MOSFETs. It was time for a bit of reverse engineering. Under the hood of the original T-4, the circuit had six small FETs in parallel for forward speed, and one FET for braking, with no reverse. After a day of head scratching and PCB probing with my oscilloscope,

Listing 1—This is servo control output compare 1 setup code.

```
// Output compare pin setup
RPOR3bits.RP6R = 18; // Make Pin RP6(RB6) OC1
OC1CONbits.OCTSEL = 0; // Use Timer 2 data
OC1CONbits.OCM2 = 1; // Use PWM mode
OC1CONbits.OCM1 = 1; // OCFA fault detection disabled
OC1CONbits.OCM0 = 0;
OC1R = 0x0400;
OC1RS = 0x0c00; // Set timer compare value for servo center position
IFS0bits.OC1IF = 0; // Clear interrupt flag
IEC0bits.OC1IE = 1; // Enable the interrupt

// Timer 2 setup
TMR2 = 0; // Clear the timer
PR2 = 0x9000; // Set the servo update time for 20 ms
T2CONbits.TCKPS1 = 0; // Set the timer prescale 1:8
T2CONbits.TCKPS0 = 1;
T2CONbits.T32 = 0; // Set the timer for 16 bit mode
IFS0bits.T2IF = 0; // Clear interrupt flag
IEC0bits.T2IE = 1; // Enable the interrupt
T2CONbits.TON = 1; // Turn the timer on
```



**Photo 3**—This is a PC control program joystick setup and calibration form.

I determined that I could remake this controller into a full-bridge speed controller without too much effort. I carved some custom heatsinks for the new International Rectifier IRFP048 MOSFETs from an old Pentium heatsink and added a cooling fan on the outside of the box to complete the new speed control system.

Three stacks of standard NiCd RC car packs power the robot. The first supply is for the main drive motors. It is a parallel stack of two 7.2-V packs. Those enable the robot to run for about 30 min. at slow speeds before needing a charge. The second supply is for the router and the PIC/WIZnet electronics. I didn't want the control electronics and motors to share the same battery packs for a reason. I knew the motor batteries would most likely be the first to run down, and I still wanted control over the servos at all times. If you've ever had an RC car or plane "run away" because the servo power went dead before the drive power, you know what I mean. I also didn't want to worry about conducted EMI from the motors into the other power supplies. The Netgear router requires 12 V, and the rest of the control electronics need 5 VDC. I used a stack of three 7.2-V NiCd packs in series with two LM350T regulator circuits to supply these voltages.

The addition of the webcam and two more servos was too much current draw for the 5-V supply because the camera alone requires 2 A. So, I

constructed an additional 5-V supply from two 9.6-V RC car NiCd packs and another LM350T regulator circuit similar to the one for the PIC24FJ64GA002. A smart charging station with temperature probes and maybe some Lithium polymer battery packs will have to go on my WiFi-PIC-Bot "upgrade-someday" list. **Photo 2** shows the inside of the control box with some of the battery packs, speed control, PIC24FJ64GA002 PCB, and WIZ810MJ PCB installed. Cooling fans are mounted on the sides of the box for the power supplies and speed control

circuits and run off of the 12-V supply. A switch for the speed control along with switches for the power supplies and fans are also mounted on the sides of the box. I left a connector inside to disconnect the motor supply batteries while I fiddle around with software and such so the robot won't drive off the table unexpectedly from a code error. A table-top robot hoist will also have to go on the to-do list. I left the original RC connectors on the speed control and steering servos so I could still drive the robot around with the RC transmitter and receiver and test the speed control by itself, if needed.

## EMBEDDED CONTROLLER

The embedded Ethernet board, speed control, and servos for controlling

steering, camera pan, and tilt are wired to a Microchip DM300027 16-bit, 28-pin development board (see **Figure 1**). I replaced the original crystal with an 8-MHz crystal to gain some speed and installed a PIC24FJ64GA002 processor. The WIZnet board is connected to the DM300027 development board by wire wrapping the header connector pins directly. The two boards communicate via the SPI bus. RB10 is used as the slave select pin. RB5 and RB6 are the PWM outputs from OC1. OC2 is used for throttle and steering servo control. OC3 and OC4 are used for camera pan and tilt servo control. I also wired RB3 with a resistor divider (10:1) and a 3.3-V Zener clamp for a battery voltage monitor. The resistor divider is required because the three battery packs in series at full charge can be up to 27 V.

## SOFTWARE

There are two separate software packages in this system. The first is the embedded code for the PIC, and the second is the UDP client on the external laptop. The RC servos operate from pulses produced by the PIC24FJ64GA002's timers. They are available on the output compare register pins. The published RC servo pulse width specification is 1 to 2 ms, although I've found that some servos need a larger range of pulse widths for control of their full span capability. If you send a pulse that is too small or too large, however, some servos will keep their internal motors on and

**Listing 2**—This is the main loop timer routine (executed every 250 ms).

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
  LJoyEx.dwSize:= sizeof(LJoyEx);
  LJoyEx.dwFlags:= LJoyEx.dwFlags and JOY_RETURNALL;
  joygetpos(JoystickID,@LJoy);
  joyGetPosEx(JoystickID,@LJoyEx);
  UpdateJoystick;
  UpdateCalibration;
  UpdateTelemetry;
end;
```

**Listing 3**—This code is for servo axis data calibration, scaling, and inversion.

```
if checkbox9.Checked=True then
  SteeringPercent:=100-(trunc(SteeringValue/(StMax-StMin)*100))
else
  SteeringPercent:=trunc(SteeringValue/(StMax-StMin)*100);
```

eventually cause internal damage to the motor or gears, so keep that in mind when writing RC servo code. If

the servo is humming, but not moving, you may have gone too far. The servo control pulse must also be

repeated every 20 ms for the servo control IC to stay awake.

Controlling one servo is rather simple. But when you have several servos, the timer interrupts can conflict, so it's best to have one timer interrupt routine resetting all the others for the main 20-ms loop. An oscilloscope can be handy for checking pulse widths and timing. The PIC24FJ64GA002 has five internal 16-bit timers. Section 14 of the document, "PIC24 Family Reference Manual," shows all of the registers involved with the timers. This microcontroller has programmable I/O pins, so you must also tell the device what pin is to be used as what function during code initialization. The set-up code for one of the servo control output pins and the associated timer registers is in Listing 1.

### WIZnet & Wi-Fi

Controlling the PICBot is achieved by sending UDP packets back and forth from a remote laptop to a WGR614 wireless router. The Linksys Internet camera and the WIZnet PCB

**Listing 4**—This is the UDP packet format and transmit routine.

```
if (UDPLinkActive=True) then
  begin
    UDPString[1]:= char(170); //10101010;
    UDPString[2]:= char(SteeringPercent);
    UDPString[3]:= chr(ThrottlePercent);
    UDPString[4]:= chr(PanPercent);
    UDPString[5]:= chr(TiltPercent);
    UDPString[6]:= char(85); //01010101
    IdUDPCliet1.Send(UDPString);
  end;
```

**Listing 5**—Receiving battery voltage telemetry from the robot.

```
if (UDPLinkActive = True) then
  begin
    VBATT_Value:=IdUDPCliet1.ReceiveString(IdTimeoutDefault);
    if (VBATT_Value[1]='V') then
      begin
        Vbyte1:=byte(VBATT_Value[6]);
        Vbyte2:=byte(VBATT_Value[7]);
        VBATT_Number:=3.3*((Vbyte1*256)+Vbyte2)/1024;
        VBATT.Text:=format('%2.1f VDC',[VBATT_Number]);
      end;
    end
  else
    VBATT.Text:='No Data';
  end;
```

# USB WITH NO STRESS

DESIGNING OF ELECTRONIC DEVICES \* CNC MACHINING(CAD/CAM) \* COMPOUND OF ELECTRONIC COMPONENTS \* DESIGNING AND MANUFACTURING OF PCB \* CONTRACT MANUFACTURING SMT & THT \* SOFTWARE DEVELOPMENT \* EVALUATION BOARDS \* KITS

PROPOX presents various series of low-cost integrated modules for data transmission via USB interface. Easy steps for projects that should be done in the short time. HOSTS, SLAVES, HUBS available.

**FEATURES OF AVAILABLE SOLUTIONS**

- \* Single chip embedded USB host/slave controller **MMusbVNC1L**
- \* Dual Port TX Buffer Dual Port RX Buffer **MMusb232RL**
- \* USB 2.0 High Speed (480Mbps/Second) compatible **MMusbX232**
- \* On-chip SIE and USB transceivers **MMusbSL811**
- \* synchronous serial protocol (USB to JTAG, I2C, SPI or bit-bang) **MMusbX232**
- \* 1.8V (chip core) and +3.3V I/O interfacing (+5V Tolerant) **MMusbX232**
- \* Two independent USB 2.0 Low speed/Full speed USB ports **MMusbVNC1L**
- \* Fast Opto-Isolated Serial Interface Mode option **MMusb2232**
- \* Integrated Level Converter on UART interface and control signals **MMusb232**
- \* Improved Power Management control for USB Bus Powered **MMusb232RL**
- \* UHCI/OHCI/EHCI host controller compatible **MMusbX232**
- \* USB Host/Slave with a standard microprocessor bus interface **MMusbSL811**
- \* USB Dual Channel Serial/Parallel Ports **MMusb2232**
- \* 300-3m Baud (TTL), 1M Baud (RS232), 3M Baud (RS422/RS485) **MMusb232**
- \* Auto-address increment mode, saves memory READ/WRITE cycles **MMusbSL811**

USA (609)-323-7568 EUROPE (0048)-58-712-8058

www.proprox.com

## Factory on Your Fingertip

From Idea to Reality

- > Design
- > Prototyping
- > Production

See Our Differences in

- > Quality
- > Service
- > Price

- \*Milling \*Turning \*Grinding \*CNC
- \*Wire Cutting \*Laser \*Plasma
- \*Water Jet \*Plastic Injection
- \*Sheet Metal \*Gear \*SLA
- \*FDM \*SLS \*LOM

# CLICK TO MAKE

**MachinePIER**

Tel: 408-421-9840 Email: sales@machinepier.com  
 Website: http://www.machinepier.com

are connected to the router using short Cat 5 cables. The router is also used as the DNS server, even though the camera could operate as its own Wi-Fi server, if needed. The UDP datagram is arranged in the UDP Datagram.doc file posted on the *Circuit Cellar* FTP site.

When the WIZnet board receives a UDP message, the PIC24FJ64GA002 first checks byte 1 and byte 6 to determine if the packet is valid. It then adjusts each servo pin timer value corresponding to the associated PWM channel. The steering and throttle timer values give the correct range of servo movement span from 1,472 to 4,672 (0x05C0 to 0x1240 hex) for a total of 3,200 counts. So, the equation for the steering servo timer value is ((percent × 32) + zero offset) and the equivalent PIC24FJ64GA002 code would be  $OC1RS = ((Rx\_Buffer[1] \times 32) + 0x05C0)$ .

## PC SOFTWARE

The UDP client software for the laptop control is written in Borland Delphi Pascal. The code uses the Microsoft multimedia library routines in Mmsystem.dll to poll the joystick driver for available joysticks and their parameter tables. I have collected several joysticks over the years, and all of them worked for the robot until I added the camera pan and tilt servos. Not all joysticks have four axis controls; however, the controller for my RealFlight simulator has five axes and several switches with a convenient USB connector. I've found that the older analog joysticks were a bit noisy for robot control, and it's just about impossible to find a laptop with an analog game port on it these days. I haven't used any of the joystick switches to control anything yet, but I can imagine the next robot upgrade will have something to do with lasers or maybe some bottle rocket launchers or Nerf guns (safer for work).

Once a joystick is selected from the main set-up form, the controls for various servos are mapped to the output channels, inverted if

**Listing 6**—This code is used to open an Internet browser window to show the robot camera video.

```
if (panel4.Visible = False) then
begin
panel4.Visible :=True;
Webbrowser1.Visible := True;
Button2.Caption := 'Click for Setup';
OpenDialog1.Title:= 'Open BotCam HTML File:';
OpenDialog1.FileName:= 'capperbotcam2.html';
If OpenDialog1.Execute then
WebBrowser1.Navigate('file://'+ OpenDialog1.FileName)
else
WebBrowser1.Navigate('about:Error Opening File');
end
end;
```

required, and calculated/calibrated from the raw data to obtain a set of 0–100 percent values (see [Photo 3](#)). The destination IP address and port is selected to match the WIZnet configuration. When the Link button is activated, UDP datagrams are sent every 250 ms with all of the data to control the robot servos. At the same time, an HTML page is loaded into an integrated browser panel that polls the camera for a real-time video stream. The camera's IP address and video setup information is located in the HTML file.

In [Listing 2](#), the first four lines poll the Microsoft multimedia driver for the current status of all the potentiometers and switches on the selected joystick. The UpdateJoystick procedure then updates the raw data to each servo control variable and shows the status of the joystick buttons and switches. The raw joystick axis values are displayed with edit

boxes, and the status of the joystick switches are shown using checkboxes. The mapping of joystick axis to the robot control servo variable is done with up/down spin buttons and a case statement.

The UpdateCalibration procedure takes this raw data and changes it into percentage values for the UDP client (inverted if needed). If the client is active, it sends the data out over the network to the robot at the designated IP address and port (see [Listing 3](#) and [Listing 4](#)). The UpdateTelemetry procedure checks for any data sent back from the robot (currently just one battery voltage) and refreshes the edit box for that value (see [Listing 5](#)).

Just as important as having an oscilloscope handy for diagnosing electrical hardware issues, a good network analyzer is a must for watching packet data. Ethereal (now called Wireshark) is a GNU open-source protocol analyzer program that works great for watching every byte that slips into and out of your network ports.

The camera video is shown on a panel that pops up over the joystick settings panel when you press the Click for BotCam button at the top of the form (see [Photo 4](#)). A file-open dialog box is used to pick the HTML file to use for the camera panel (see [Listing 6](#)).

The HTML file downloads and runs a JavaScript file from the camera that streams video at a maximum of 30 frames per second at a maximum size of 640 × 480 pixels. The fun part is that I can



**Photo 4**—This is a runtime form with a PICBotCam video screen.



# 3 PORT INTERFACE RS-485 to Ethernet Converter



**Only**  
**\$170**



**RS-485 to Ethernet Converter**

---

## Powerful feature

---

- Protocol converter RS485 between Ethernet
  - Offer TCP/IP Communication to Devices with RS485 I/F
- 

## Specification

|                        |  |   |
|------------------------|--|---|
| <b>Network</b>         | : TCP, UDP, DHCP, ICMP, IPv4, ARP, IGMP, PPPoE, Ethernet, Auto MDI/MDIX , 10/100 Base-TX Auto negotiation (Full/half Duplex) |   |
| <b>Serial</b>          | : RS485 3 Ports, 1,200~115,200 bps, Terminal block I/F Type  |   |
| <b>Control program</b> | : IP Address & port setting, serial condition configuration, Data transmit Monitoring  |   |
| <b>Accessory</b>       | : Power adapter 9V 1500mA, LAN cable   |   |
| <b>Etc</b>             | : - DIP Switch(485 Baud Rate setting)  | - LED: Power, Network, 485 Port transmission signal |

let anyone on the 'Net log in and watch the video while the robot is running. I just need to give a viewer the IP address and log-in credentials. I can also set the Wi-Fi security to keep hackers out.

## AM I DONE YET?

No way! There's a never-ending list of gadgets and features for a "Future-Bot." Now that the robot has a camera, it needs some sort of VoIP to communicate with the people it encounters during its travels. Have I mentioned all of the control buttons that need to control something? A high-resolution (approximately 8 megapixels) still camera could be useful for getting a good shot of something the Internet camera finds interesting. Maybe I'll rent it out to the fire department for search-and-rescue operations in tiny crawl spaces. It could putz up and down the beach with an attached metal detector and tiny treasure scoop. I need a job at NASA so I can do this full time.

I hope you enjoyed reading about the PICBot. Have fun building your own. (I know you want one.) ☑

Scott Coppersmith ([rscopper@aol.com](mailto:rscopper@aol.com)) holds a BSEE from Michigan Technological University and is currently working as a senior engineer for Robert Bosch LLC. He also teaches evening classes at Ivy Tech Community College in South Bend, IN. Scott's hobbies include Tesla coils, fusors, lasers, embedded systems, and Delphi programming.

## PROJECT FILES

To download code, go to [ftp://ftp.circuitcellar.com/pub/Circuit\\_Cellar/2009/224](ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2009/224).

## RESOURCE

Microchip Technology, Inc., "PIC24 Family Reference Manual," DS39704A, 2006.

## SOURCES

### IRFP048 MOSFETs

International Rectifier | [www.irf.com](http://www.irf.com)

### WVC54GC Wireless Internet video camera

Linksys | [www.linksys.com](http://www.linksys.com)

### DM300027 Development board and PIC24FJ64GA002 microcontroller

Microchip Technology, Inc. | [www.microchip.com](http://www.microchip.com)

### WGR614 Wireless router

Netgear | [www.netgear.com](http://www.netgear.com)

### WIZ810MJ Network module

WIZnet, Inc. | <http://wiznet.co.kr/en/>

## Expand Your I/O with Rabbit® RIO

### Versatile Device with I/O Options

- Add functionality without costly processor changes
- Multiple Processor Interfaces
- Add 38 I/O
- Configure I/O for PWM, TRIAC, input capture, or decoder



Rabbit RIO™ Programmable I/O Application Kit for **\$299**

**RABBIT**™



Order Online At [rsappkits.com](http://rsappkits.com)

08033

 **Electronicstalk**

} Keep abreast of the latest news from your industry, delivered **free** to your desktop

Sign-up today at [www.electronicstalk.com](http://www.electronicstalk.com)

With a library of more than 49,000 articles from more than 3,000 companies, we are the number one destination for people making purchasing decisions!

Electronicstalk matching buyers with sellers  
[www.electronicstalk.com](http://www.electronicstalk.com)



# ROBO 2009

## Business

CONFERENCE & EXPO

APRIL 15-16, 2009

HYNES CONVENTION CENTER

BOSTON, MASSACHUSETTS  
WWW.ROBOBUSINESS.COM

JOIN THESE LEADING COMPANIES AT 2009 ROBOBUSINESS

Produced By



Robotics Trends

ROBOTICS BUSINESS REVIEW

Founding Sponsor

**iRobot**

Premier Sponsor



THE TECHNOLOGY COLLABORATIVE  
Accelerating Digital & Robotic Innovation.

Silver Sponsors



Holland+Knight



Academic Sponsor



Association Sponsors



Media Co-Sponsors



Listing as of Jan. 30. For a current list of Sponsors & Exhibiting Companies, visit [www.robobusiness.com](http://www.robobusiness.com).

**REGISTER NOW**

**SAVE \$300**

**ON YOUR FULL CONFERENCE PASS!**

USE PRIORITY CODE **RBCC**

## BUILDING THE ROBOTICS INDUSTRY

Now in its sixth year, **RoboBusiness Conference & Exposition** is the must-attend event for those involved in the business and technical issues related to the development of the personal, service and mobile robotics industry. The emphasis of RoboBusiness is on what technologies and applications are emerging with the greatest opportunity to be commercially successful.

### WORLD CLASS CONFERENCE

**RoboBusiness 2009** features two days of keynotes and general sessions delivered by internationally recognized leaders from business, government and academia. RoboBusiness conference tracks:

- Business Development and Investment
- Technology and Standards
- Security, Defense and First Responder Robotics
- Consumer Robotics
- RoboMedicus - Healthcare Robotics
- Autonomy and Mobility

### EXPOSITION & SPECIAL EVENTS

- Evening Networking Reception Sponsored by **iRobot**
- Speaker Meet & Greet
- Birds-of-a-Feather Discussions
- Expo floor featuring dozens of leading companies from around the world

**FOR COMPLETE EVENT DETAILS VISIT [WWW.ROBOBUSINESS.COM](http://WWW.ROBOBUSINESS.COM) OR CALL 800-305-0634**

RoboBusiness is a professional, trade event. All attendees must be at least 18 years old to attend.

For Information on Sponsorship and Exhibiting Opportunities, contact Ellen Cotton at [ecotton@ehpub.com](mailto:ecotton@ehpub.com) or 508-663-1500 x240



# MP3P DIY KIT, Do it yourself

(Include Firmware Full source Code, Schematic)

## • myPIC



Only

**\$160**   **\$220**  
qty 100   qty 1

## • myWave (MP3 DIY KIT SD card Interface)



Only

**\$150**  
qty 100  
**\$200**  
qty 1

## • myAudio (MP3 DIY KIT IDE)

Only

**\$180**  
qty 100  
**\$220**  
qty 1



### Powerful feature

- MP3 Encoding, Real time decoding (320Kbps)
- Free charge MPLAB C-Compiler student-edition apply
- Spectrum Analyzer
- Application: Focusing for evaluation based on PIC
- Offer full source code, schematic

### Specification

Microchip dsPIC33FJ256GP710 / 16-bit, 40MIPs DSC  
VLSI Solution VS1033 MP3 CODEC  
NXP UDA1330 Stereo Audio DAC  
Texas Instrument TPA6110A2 Headphone Amp(150mW)  
320x240 TFT LCD  
Touch screen  
SD/SDHC/MMC Card  
External extension port (UART, SPI, I2C, I2S)

### Powerful feature

- Play, MP3 Information, Reward, forward, Vol+/-
- Focusing for MP3 Player
- SD Card interface
- Power: battery
- offer full source code, schematic

| Item          | Specification                                   |
|---------------|---|
| MCU           | Atmel ATmega128L                                |
| MP3 Decoder   | VS1002 / VS1003(WMA)                            |
| IDE Interface | Standard IDE type HDD(2.5", 3.5")               |
| Power         | 12V, 1.5A                                       |
| LCD           | 128 x 64 Graphic LCD                            |
| Etc           | Firmware download/update with AVR ISP connector |

### Powerful feature

- Play, MP3 Information, Reward, forward, Vol+/-
- Focusing for full MP3 Player (Without case)
- IDE Interface
- Power: Adapter
- Offer full source code, schematic



## FAT File System Review (Part 1)

### Open Files and Perform Operations

If you are designing a system with a removable memory device, try using a file storage system such as the FAT file system. George describes how to open files with the system, use a CompactFlash card, and perform other basic operations.

For the past two years, we've been exploring the C language and specifically how to use it in embedded systems. I've introduced you to the language and described how to use it to construct typical systems. I've also described how to include more aspects of a design such as flowcharts, UML diagrams, and other development tools. Well, it's time to move on from the lower-level details of the language and move up a level or two, although from time to time I'll revisit the basics. Feel free to e-mail me if you have questions about embedded C systems.

#### FAT FILE SYSTEM

If you're planning an embedded system, it's probably not going to be a simple design with just a push button as an input and an LED as an output. In today's environment, it's going to be rather complicated and full of features. How in the heck do you design such a system? I will answer that question indirectly because I don't believe there is a simple single answer. There are families of answers (or approaches) to the design. My goal is to present you with several alternatives. Your goal is to understand each one and decide how they might best fit your application.

##### Listing 1—Basic C file operations

```
FILE *fopen(const char *path, const char *mode);
int fclose(FILE *fp);
int fgetc(FILE *fp);           // read one character
int fputc(int c, FILE *fp);   // write one character
```

If you've been reading *Circuit Cellar* for the past few years, you may remember reading articles about the FAT file system. A few of the articles are listed at the end of this article. In fact, Jeff Bachiochi recently completed a two-part series on the topic ("Access SD Memory Cards," *Circuit Cellar* 222 and 223, 2009). For many of today's embedded systems, it's likely that one feature on the requirements list is a removable memory device. Years ago, we actually put EPROMs on plug-in memory cards. (Remember the early video games?) Today you would either use a CompactFlash card or a MultiMediaCard (SD/MMC). To meet that requirement, you need a hardware interface to the card, and you have to design your own file system or use an existing file system.

If you want to design your own file system, good luck. You've got a lot of work ahead of you and you'll need it. If, on the other hand, you want to leverage the PC with Windows, MAC with OS X, and other off-the-shelf computers (some using Linux), then look into using an existing file system.

With DOS, Microsoft created a file system, the same system that first appeared on floppy disks and evolved onto hard disks. As disk capacity grew, the file system changed to make room for larger capacity disks. That file system is referred to as FAT, FAT12, FAT16, and FAT32. Somewhere in the mix of designs, long file names were included. (Refer to the Resources at the end of the article for information about the file systems.)

If you purchase a CF or SD/MMC

**Listing 2**—Alternative C file operations

```
char* fgets(char *string, int length, FILE fp);
int fputs(const char *string, FILE *fp);
```

memory card, it will come preformatted in one of the FAT file systems. The problem you are going to run into using the off-the-shelf cards is the same problem you had with EPROM memory devices. The size of the devices keeps growing and the one you designed with is no longer available or is available only at a higher cost. Try to purchase a 1-KB EPROM (or even a 32-KB device). If you need only 32 MB on a memory card, that's great today. But someday you'll find that size is no longer manufactured. As the device card grows, an older file system won't be able to access the entire available memory. And changing file systems is a big task.

It looks like I've come up with more issues than answers (or even the hope of answers). So, let's get into the next part: "What are we going to do?" From the top of the design pile, if you look at the C language, you'll find it supports file I/O. You'll find a good description at [http://en.wikipedia.org/wiki/C\\_file\\_input/output](http://en.wikipedia.org/wiki/C_file_input/output).

### FILE OPS & PROTOTYPES

The most basic file operations are open, close, read, and write. Their prototypes in C are in [Listing 1](#).

There are alternatives to the basic reading and writing of one character at a time. One common pair of procedures is reading and writing a string (see [Listing 2](#)).

By now, you should be starting to understand what these prototypes mean. The new construct in all of these prototypes is the `FILE *fp`. It is a pointer to a structure that can be seen in [Listing 3](#).

Every time you open a file using C, you are creating a structure and filling in the details. The structure is all that is needed to perform all the file operations that the C language has to offer. This is a simple straightforward interface, and you should be suspicious that it's too easy. I suspect that there is a lot of code to support the simplicity of this interface. For example, `*buffer` is a pointer to a character buffer. What's up with that? Where is that buffer stored and how big is it? A useful assignment that would give you a lot of insight into this problem is to find and review the Linux file I/O code.

### OPEN A FILE

Let's examine how to open a file. The `fopen` procedure takes two parameters. The first is the complete file name including the path. So `E:\Data\CCI\Code\Test\TestFile.txt` is a string that can be used as a complete reference to the file. Note that it's a string and you pass a pointer to that string. If you are running C on a PC, the OS takes care of translating that path and file name into a specific hardware location. If you're designing an embedded system, you're responsible for that translation. The second parameter is the mode that you are using to open the file. Different modes are "r" (read), "w" (write), and "a" (append). With these and other characters, you can specify operations, such as append, start from the beginning, and open for both reading and writing.

The return parameter from the `fopen` procedure is a pointer to the `FILE` data structure. If the pointer is `NULL` (a C-defined constant), the

**Listing 3**—Pointers pointing to 8-bit data in 16-bit elements

```
typedef struct {
    int          level;           // fill/empty level of buffer
    unsigned    flags;          // File status flags
    char        fd;             // File descriptor
    unsigned char hold;         // Ungetc char if no buffer
    int         bsize;          // Buffer size
    unsigned char *buffer;      // Data transfer buffer
    unsigned char *curp;        // Current active pointer
    unsigned    istemp;         // Temporary file indicator
    short       token;          // Used for validity checking
} FILE;
```

**Listing 4**—The lowest level CF interface routines

```
INT16    Init_CF(void);

INT16    CFCheck_RDY(void);
INT16    Wait_RDY(void);
INT16    CFCheck_RDY_ALT(void);
INT16    Wait_RDY_ALT(void);

UINT16   ReadRegB(UINT16 *addr);
UINT16   ReadRegW(UINT16 *addr);
UINT16   WriteRegW(UINT16 *addr);
UINT16   WriteRegB(UINT16 *addr);

void     ReadCF_SectorB(UINT8 *dest, UINT32 LBAlocation);
void     ReadCF_SectorW(UINT16 *dest, UINT32 LBAlocation);
void     WriteCF_SectorB(UINT8 *dest, UINT32 LBAlocation);
void     WriteCF_SectorW(UINT16 *dest, UINT32 LBAlocation);

void     ReadIdentityInfoW(UINT16 *buff);
void     ReadIdentityInfoB(UINT16 *buff);

UINT16   CFByteSwap(UINT16 data);
UINT8    Get8Bits(UINT8 *Buffer, UINT16 addr);
UINT16   Get16Bits(UINT8 *Buffer, UINT16 addr);
UINT32   Get32Bits(UINT8 *Buffer, UINT16 addr);
```

command failed to open the file. Aren't you glad you read and paid close attention to my articles on structures and pointers! (Refer to *Circuit Cellar* issues 198, 200, 202, 204, 206, 208, 210, 212, and 214.)

I am going to skip over the middle of the FAT file system design and examine the lowest level interface to the hardware. It's the only other piece of information that's solid at this point. Again using the Internet, I came up with several links to CompactFlash information.

## CompactFlash

There are several ways to design a hardware interface around a CF card. One simple method has eight registers (using three address lines) that provide commands, data, and status information. Also, either an 8- or 16-bit data interface is available. The hardware interfacing to an SD/MMC card basically uses the serial peripheral interface (SPI) format. I have not yet done this type of design work, so I'll say no more at this point about SD/MMC card interfacing. Again, refer to Jeff's articles for details about an SD hardware interface.

The lowest level of CompactFlash hardware and software interface can be accomplished in the routines in

**Listing 4.** The `Init_CF()` routine grounds the RESET pin on the CompactFlash card. Note that not all cards respond correctly or completely to this reset. Some require that power actually be removed. Some designs have a high-side FET actually disconnecting the power supply. So look out.

The next four routines—`CFCheck_RDY()`, `Wait_RDY()`, `CFCheck_RDY_ALT()`, and `Wait_RDY_ALT()`—look at a ready or alternate ready signal from the CF card. This signal is the reply that indicates that the CF card has completed the previous operation. Sometimes you just want to look at the signal (the check procedures); other times, you want to wait until the previous operation has completed (the wait procedures). Be careful. You will hang the system if you are waiting for a ready signal that never materializes. This probably is not a good situation in an embedded system.

The next four routines—`ReadRegB()`, `ReadRegW()`, `WriteRegW()`, and `WriteRegB()`—all read and write to a register in the CF card. Again, check the literature about CompactFlash cards. The CF hardware interface provides for an 8- or 16-bit data path. If you've got a smaller system,

an 8-bit interface might make the most sense. If, on the other hand, you're looking for maximum performance, the 16-bit data path is available.

The next four routines—`ReadCF_SectorB()`, `ReadCF_SectorW()`, `WriteCF_SectorB()`, and `WriteCF_SectorW()`—read from and write to a sector on the CF card. The basic information is accessed in  $512 \times 8$  bit or  $256 \times 16$  bit blocks. A special routine `ReadIdentityInfoW()` reads words from a specific card location to get the identification information about the CF card. I included the prototype for reading bytes `ReadIdentityInfoB()`, but I didn't code that one because I have a 16-bit interface.

The next four routines—`CFByteSwap()`, `Get8Bits()`, `Get16Bits()`, and `Get32Bits()`—deal with manipulating the data that has been read from the card. `CFByteSwap` deals with the Big Endian/Little Endian issues. This concerns the byte order used when saving data. Murphy's law says it won't be what you want it to be. If these Endian issues are a new topic to you, please look them up. The `GetnBits()` routines return the specified amount of data from the buffer. Different parameters are saved as 8, 16, or 32 bits for efficiency. They are packed with no padding between parameters. Any size parameter could start at any memory location.

The `LBAlocation` is a 32-bit number specifying the location (address) on the CF card. Disk drives use heads, cylinders, tracks, and sectors to get to a specific location on a drive. The CF card uses a formula that equates the `LBAlocation` to a head, cylinder, or sector notation. The `LBAlocation` is the head (8 bits), cylinder (16 bits), and sector (8 bits), all combined into a 32-bit number.

## APPLICATION REQUIREMENTS

I think we've got both the high-level and the low-level ends of the design covered. All you have to do next is fill in the middle. I originally wanted to write this article as a call for a FAT file system design (sort of

a “roll our own” project). But I came to my senses and decided this was too big a project to tackle. There are so many CF cards and I should also deal with SD and MMC cards. Also, each application has its own resources and requirements. One such requirement is how many files can be open. Each open file has a FILE structure associated with it and probably buffers for data. If you’re building a simple system, or just using the card for program updates, then a fancy file system makes no sense. On the other hand, if you’ve got one of the new ARM CPUs with an Ethernet interface built in (all for about \$5) and lots of system memory, then you would want and could use a much more robust file system.

Let’s explore the FAT file system in more detail, as designed by Microsoft and extended by the card manufacturers and others. The Wikipedia reference is a good condensed explanation of the different FAT versions. The original FAT, sometimes referred to as FAT12, is probably old enough that you don’t need to support it. The FAT16 version supports volumes up to 2 GB, and that’s probably sufficient for today’s designs. But when will you no longer be able to purchase 2-GB CF cards? When will the larger capacity cards be less expensive? It probably won’t happen next year, but it probably will in the next four to five years. So designing a system compatible with FAT16 has a shelf life and you better consider FAT32 as a real possibility. Why not design or plan for both?

Another consideration is long file name support. This is one decision that’s clearly application-dependent. The systems I’m working on can require all file names to be in the 8.3 format (eight characters for the name and three for the extension). And also any directory names can be limited to eight characters. If your system can’t meet these requirements, you need to include support for long file names.

The last major consideration is the requirement that the CF or

SD/MMC card format be compatible with another computer system. A straightforward example is that the card be readable and writable in your embedded system and in a PC. Then the questions become: Which OS is the PC running? Does that OS support the same file system as your embedded system does? If you are going to read and write memory cards only in your system, compatibility with an existing OS is not an issue.

## WORKING WITH THE SYSTEM

So how does one work with the FAT file system? I urge you to search the ‘Net and get as much information as you can stand. And there is a lot available. You’ll find that you need to detect and initialize the card, read the identity information,

and build the information that you will use to access the card from the identity information.

Jeff described this process for an SD card. In the files section of this article, I took code from a 2005 *Circuit Cellar* article by Ivan Sham, William Hue, and Pete Rizun and passed it through SourcePublisher from Scientific Toolworks (“Portable FAT Library for MCU Applications,” *Circuit Cellar* 176, 2005). I used this program to document my work and help organize other C code. I included the original files and the output from SourcePublisher. Together with these links and Jeff’s most recent columns, you now have an introduction to a file system.

But wait. There’s more. Next time, I’ll get into the heart of some C code for a file system. 📄

*George Martin (gmm50@att.net) began his career in the aerospace industry in 1969. After five years at a real job, he set out on his own and co-founded a design and manufacturing firm (www.embedded-designer.com). His designs typically include servo-motion control, graphical input and output, data acquisition, and remote control systems. George is a charter member of the Ciarcia Design Works Team. He’s currently working on a mobile communications system that announces highway info. He is also a nationally ranked revolver shooter.*

## PROJECT FILES

To download code, go to [ftp://ftp.circuitcellar.com/pub/Circuit\\_Cellar/2009/224](ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2009/224).

## RESOURCES

CompactFlash Association, “CF+ and CompactFlash Specification Revision 2.0,” 2003.

Compu Phase, “Implementing FAT on CompactFlash cards, SD/MMC cards and USB sticks,” 2008, [www.compuphase.com/mbr\\_fat.htm](http://www.compuphase.com/mbr_fat.htm).

Microsoft Corp., “Microsoft Extensible Firmware Initiative FAT32 File System Specification,” 2000.

C. Quirke, “Understanding FAT,” <http://users.iafrica.com/c/cq/cquirke/fat.htm>.

M. Samuels, “PIC a CompactFlash Card,” *Circuit Cellar Online* 127, 2001.

I. Sham, W. Hue, and P. Rizun, “Portable FAT Library for MCU Applications,” *Circuit Cellar* 176, 2005.

Wikipedia, “CompactFlash,” <http://en.wikipedia.org/wiki/CompactFlash>.

———, “File Allocation Table,” [http://en.wikipedia.org/wiki/File\\_Allocation\\_Table](http://en.wikipedia.org/wiki/File_Allocation_Table).

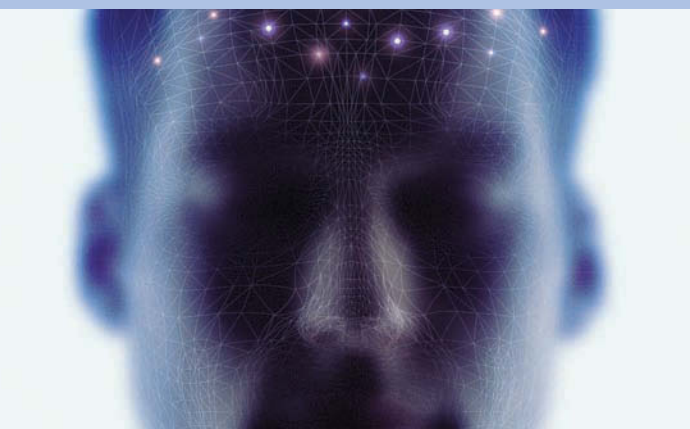


# sensors expo & conference

Conference: June 8 - June 10, 2009  
Exhibits: June 9 - June 10, 2009  
Donald E Stephens Convention Center  
Rosemont, Illinois  
[www.sensorexpo.com](http://www.sensorexpo.com)

## Advances in Measurement, Monitoring, Detection & Control

New Approaches • New Technologies • New Applications • New Ideas



### Don't Miss the Sensors Opening Keynote



#### Cassini: Five Years at Saturn

#### Dr. Kevin Grazier

*Investigation Scientist & Science Planning Engineer, Cassini/Huygens Mission to Saturn & Titan, NASA's Jet Propulsion Laboratory (JPL)*

### This Year's Conference Program Covers 18 Tracks

- Sensor Interfaces & Sensor Integration
- Sensor Systems Design
- RF Sensing
- Wireless Sensor Networks
- Energy Harvesting
- Energy Conservation
- Low-Power Sensing
- Harsh Environments
- Position Sensing
- Fiber Optics
- Machine Health & Predictive Maintenance
- Smart Materials
- Novel Approaches to Measurement & Detection
- Environmental Monitoring
- Business Trends & Issues
- Wireless Standards
- Location-Aware Sensing
- Novel Approaches to Biodetection

**Register Today for Your Conference Pass at the Early Bird Rates!**

**Or, Sign Up Now for a FREE Expo Hall Pass!**

**Visit [www.sensorexpo.com](http://www.sensorexpo.com) or call 877-232-0132 or 972-620-3036 (Outside U.S.).**

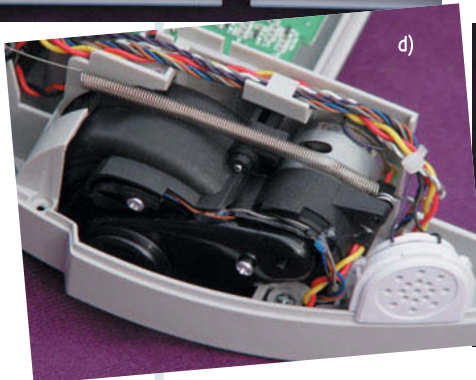
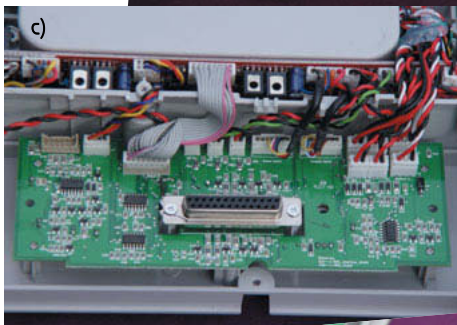
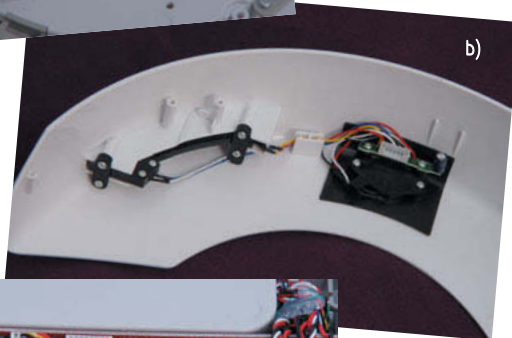
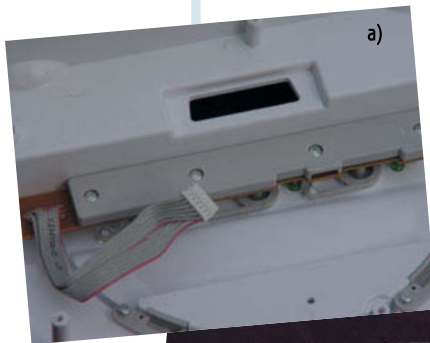
**Don't Forget to Use Your Source Code: 303M**



## Programmable Robotics (Part 1)

### Build on an Existing Robot Platform

If you want to work on a robot but don't have to time build one from scratch, check out iRobot's Create. The robot incorporates several customizable features, including a command module for on-board programming, a serial cable to send individual commands, 10 built-in demos, over 30 built-in sensors, and a 25-pin expansion port.



Occasionally, I get dragged to the nearest mall, especially around Christmas time. Few of the stores interest me. But Brookstone is different. It is a nation-wide specialty retailer with an assortment of consumer products that are functional in purpose, distinctive in quality and design, and not widely available from other retailers. Years ago, I was drawn in by a Frisbee-shaped gizmo wandering around on the floor near the store's entrance. It was an iRobot Roomba vacuuming robot. iRobot's literature described it as the first practical and affordable home robot. You may have heard of iRobot, which developed robots like the PackBot that was used to search the rubble of the World Trade Center in New York after the September 11 terrorist attacks. It is now used by the U.S. military overseas.

Apparently, iRobot quickly sold more than 1 million Roomba units. This success led to a line of similar robotic appliances used for vacuuming, sweeping, washing floors, and cleaning pools and gutters. As you can imagine, when the robots started becoming popular, hacking sites sprang up with information about ways to use these

platforms for unintended purposes.

iRobot took advantage of this growing frenzy and developed an offshoot based on the Roomba's proven technology.

**Photo 1**—The Create's Power, Play, and Advance buttons and LEDs (a) and wall sensor and IR receiver (b) along with every other sensor can be disconnected from the interface board (c). Motor gearboxes with optical encoders (d) give the Create (e) a movement resolution of 1 mm. Angle is determined to 1° of rotation based on the left and right encoder values.

The Create was born with two basic models. In this article, I will discuss what iRobot provides as a system and how you can use the technology.

## PROGRAMMABLE ROBOT

The Create mobile robot platform is for educators and developers alike. It's mobile out of the box. There is no need to assemble the drive system or worry about low-level code. The basic unit includes a serial cable for sending individual commands from a PC or writing basic scripts of up to 100 open-interface commands, which can be stored on the robot. Ten built-in demos perform various pre-programmed behaviors. More than 30 built-in sensors enable the robot to react to both internal and external events (see [Photo 1](#)). In addition to a serial communications port, there is a 25-pin expansion port, to which you can connect your own electronics (sensors, actuators, wireless connections, computers, cameras, or other hardware). The cargo bay has various threaded holes for securely mounting your own brand of hardware. You can add a fourth wheel to improve the stability of larger payloads, but this slightly reduces its ability to adapt to changes in grade.

The advanced unit includes the iRobot Command Module for on-board (autonomous) programming. The Create is fully compatible with all iRobot Roomba accessories, including rechargeable batteries, a power supply, a home base, a remote control, and virtual walls. The Command Module screws onto the 25-pin expansion port at the front of the cargo bay. It contains an Atmel ATmega168 microcontroller that communicates with the Create through a serial connection in the expansion port. It is programmed via a USB port. Let's begin with a look at the basic unit's array of peripheral I/O devices.

## UNDER THE HOOD

The Create is preprogrammed with an operating system that enables access to all of its input and output devices by passing commands via a serial communications port (default

speed of 57,600 bps). Commands include an opcode (128–159) followed by a number of data bytes, depending on the opcode. Table 1 lists the opcodes and data requirements of each. I color-coded the table so you can see the groups. Five commands place the Create into special modes.

The Create begins in Off mode and remains there until a Start command is received. The Start command places the Create into passive mode ready for additional commands. From

the passive mode, the Create accepts only the Baud, Safe, or Full commands. Safe and Full are similar commands that give you complete control using any of the other commands. Safe mode, as its name implies, has some automatic self-preservation actions, which prevent it from falling off an edge, for instance. In Full mode, if you don't pay attention to its cliff sensor, for instance, it would attempt to fly right off a ledge (thereby needlessly

## No other PCB-design tool gives you more value per dollar

Boards designed under EAGLE are developed in one-man businesses or in large industrial companies. Most of the top companies are our customers. The crucial reason for selecting EAGLE is not usually the low price, but rather the high-end functionality along with the ease of use. And EAGLE users appreciate 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!

Version 5 is even easier to use, especially for beginners, due to an enhanced user interface.



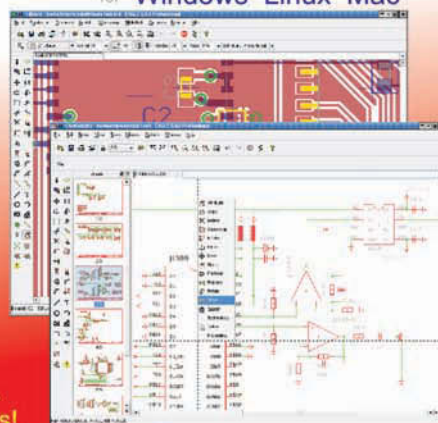
**EAGLE**  
Version 5

Schematic Capture • Board Layout  
Autorouter

for Windows® Linux® Mac®

### Version 5 Highlights

- ▶ Stand-alone schematic editor available.
- ▶ Automatic signal/contact cross references using frame coordinates.
- ▶ Right-mouse click for more consistent Windows UI.
- ▶ User-definable attributes for parts.
- ▶ Schematic sheet management.
- ▶ Hiding approved DRC and ERC errors.
- ▶ PRINT preview and text-searchable PDF output.
- ▶ Improved search engine for help.
- ▶ And much, much more.



Pick the level that is right for you — pay only the difference for upgrades!

|   | Light      | Standard   | Professional |
|---|------------|------------|--------------|
| Max. number of schematic sheets                   | 1          | 99         | 999          |
| Max. board size                                   | 4x3.2 inch | 6.4x4 inch | 64x64 inch   |
| Max. # of signal layers                           | 2          | 4          | 16           |
| Layout or Schematic Editor                        |            | \$249      | \$498        |
| Layout and Schematic Editor                       |            | \$498      | \$996        |
| Layout Editor and Autorouter                      |            | \$498      | \$996        |
| Layout Editor and Schematic Editor and Autorouter | \$49       | \$747      | \$1494       |

Standard and Light Editions have full functionality except for the limitations mentioned in the table.

You can use EAGLE Light for evaluation and non-commercial applications without charge. Download it from our web site.

[www.cadsoftusa.com](http://www.cadsoftusa.com)  
800-858-8355

CadSoft Computer, Inc., 19620 Pines Blvd., Suite 217, Pembroke Pines, FL 33029  
Hotline (954) 237 0932, Fax (954) 237 0968, E-Mail: [info@cadsoftusa.com](mailto:info@cadsoftusa.com)

Windows / Linux / Mac are registered trademarks of Microsoft Corp. / Linux / Solaris / Apple Computer, Inc.

testing its poor aerodynamic characteristics).

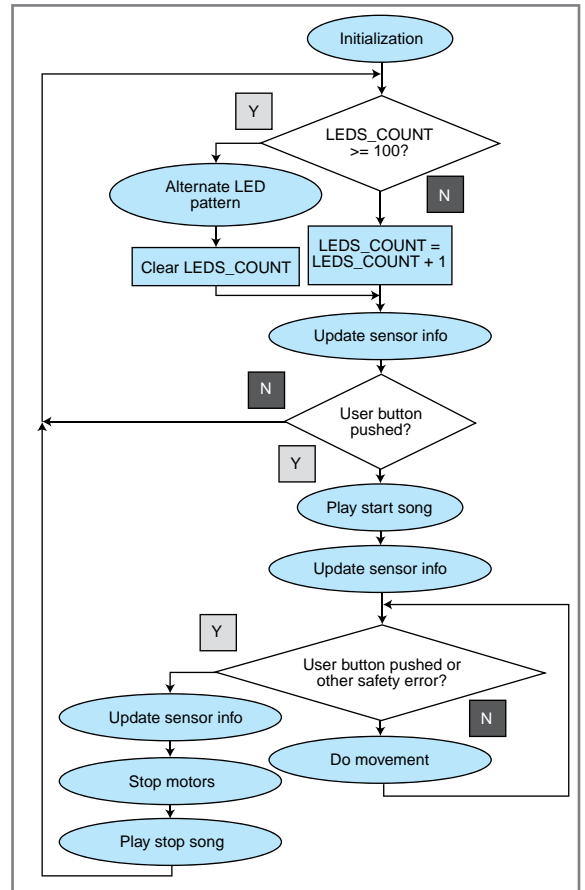
The built-in demo programs can be called through the open interface using the commands in rows 6–8 and row 15 (see Table 1). Note that while Cover, Cover and Dock, and Spot are available as individual commands, the Demo command (with a data byte of 0–9) lets you choose any of the built-in demos. For instance, the Cover and Dock demo instructs the Create to use random bounce, wall following, and spiraling to cover a room's floor area. If it sees an IR signal from its home base, it will zero in on home base, dock with it, and recharge its batteries.


This brings us to the real nitty-gritty. The commands in rows 2, 9–14, 16–22, and 26–29 have to do with the Create's I/O. The Create's outputs are left and right motors, LEDs, tone, and output bits on the 25-pin expansion connector (6 bits, three digital, and three PWM). On the input side, there are wheel drop sensors, bumpers, wall sensors, cliff sensors, virtual wall (IR)

**Figure 1**—This is the idle flowchart I wrote by reviewing the code in the example DRIVE.C on the CD that came with the Create.

sensors, overcurrent sensors (for wheel and PWMs), buttons, distance, angle, charging state, battery voltage, system current, battery temperature, battery charge, battery capacity, and inputs from the 25-pin expansion connector (four digital bits and one 10-bit analog input).

You can refer to the "Open Interface Specification" in the Resources section for more information on all of this, but I want to touch on it briefly. There are two ways to move the Create.





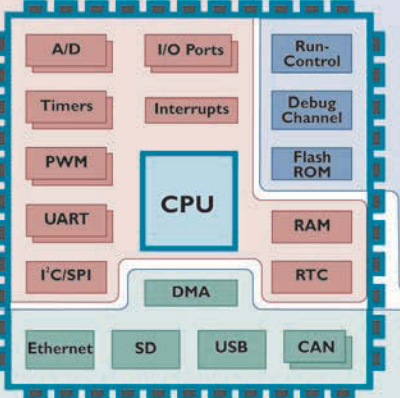
**KEIL™**  
An ARM® Company

**Software Development Tools**

**The Leader in Microcontroller Development Solutions**

**C/C++ Development Kit** including best-in-class compilers, genuine Keil μVision®, and royalty-free RTX RTOS.

**ULINK®2 Adapter** for target debugging and Flash programming.



Keil **RTOS and Middleware** components are specifically optimized for embedded systems and include TCP/IP, Flash File system, USB and CAN support.

**ARM**  
[www.keil.com/arm](http://www.keil.com/arm)

**Cx51**  
[www.keil.com/c51](http://www.keil.com/c51)

**C166**  
[www.keil.com/c166](http://www.keil.com/c166)

Out-of-the box support for more than 1,400 Microcontroller devices.

Call **1-800-348-8051** for a free demo CD.

[www.keil.com](http://www.keil.com)

The drive command requires 2 data bytes, a velocity of ±500 mm/s, and a radius of ±2,000 mm. That's a maximum speed of approximately 1 MPH! This mode is similar to driving a car using the gas pedal for speed control and the steering wheel for direction control. The drive direct command requires 2 data bytes, a right velocity of ±500 mm/s, and a left velocity of ±500 mm/s. This mode is similar to driving a bulldozer or tank using independent forward and reverse levers for each wheel.

Feedback on movement comes from the wheel encoders as values for distance (−32,768 mm to 32,767 mm) and angle (−32,768° to 32,767°). The distance is the total distance covered by each wheel's encoder divided by two. If the Create is spinning in place, its distance remains at zero. The angle is a relationship of the distances traveled by each wheel. The distance between wheels is 260 mm. If only one wheel is moving, that wheel must travel  $2\pi R$ , or 1,634 mm (i.e.,  $2 \times \pi \times 260$  mm), to get back to where it started, so  $1,634/360 = 4.5$  mm/degree.

For every 4.5 mm difference in travel distance between the two wheels, the orientation will change 1°. With anything other than point contact (of thin wheels), you can guesstimate only the actual turning radius. This will vary greatly depending on which part of a wide wheel is making contact with the floor. That's why navigation by "dead reckoning" loses accuracy over time. This could be corrected by using an electronic compass to determine direction instead of dead reckoning. Compasses aren't necessarily perfect either. So, in the big scheme of things, it's best to not put all of your eggs in one basket. This is getting beyond the scope of this article, so let me just caution you: in most cases, you can rely on the angle for immediate results (i.e., turning around), but don't depend on it to keep track of your direction over the long term (i.e., trying to set up parallel sweeps to cover a large area).

Let me make one last note about the commands in rows 23–25 (see Table 1). The Script commands enable you to record and play back a number of consecutive commands. When used in conjunction with the Wait commands, you can have the Create perform a preprogrammed sequence of events. Here is an example: wait distance (1,000 mm), wait angle (90), wait distance (1,000 mm), wait angle (90), wait distance (1,000 mm), wait angle (90), wait distance (1,000 mm), and wait angle (90), which when played back commands the Create to drive in a 1 m × 1 m square pattern.

## SERIAL TETHER

The Create can play any of the demos via push button selections after you press the Power button. However, to get the Create (basic unit) to do your bidding using the open interface commands, you need a serial link to something that can send these commands via the link. This can be done with a terminal emulator (e.g., RealTerm) running on your favorite computer with a serial interface.

While running RealTerm on your PC with the Create connected via the serial cable, you can use your keyboard to

type in commands (in decimal with spaces separating input values). The Create will execute the commands in the order they are received. If you type "128 132 139 2 0 0 <cr>," you will turn on the Play (>) LED. If you enter a drive command, support the system underneath with its wheels off the floor (full mode). The Create will most likely rip your serial cable out as it bolts across the floor unless you have a long tether.

I know what you are thinking: "Not user friendly, this tether." Well, there is of course a better way. You can eliminate the tether by supplying an RF link. I want to stop from going down this path for the remainder of this column. Instead, I want to focus on the Create's advanced unit, adding the command module to the 25-pin expansion connector in the forward end of the cargo bay.

## COMMAND MODULE

You can use a PC to enter open interface commands that the Create can receive via its serial port and execute in its operating system. The command module uses an ATmega168 flash memory microcontroller to send commands to the Create. This is accomplished by writing a control program and programming it into the microcontroller.

The control program uses the open interface commands to access the Create's functions. How is this any different than typing commands to the Create through the tethered serial port? It's not! However, the complicated answer is much more satisfying.

The most obvious difference is that the tether disappears because the Command Module rides piggyback on the Create. Second, I don't know about your typing speed (or accuracy for that matter), but the microcontroller can send commands at blazing speeds. Here comes the most important fact. Even if I could type a drive and read sensor commands fast enough, I would not be able to check all of the important sensor states and decide to stop before I hit the wall on the other side of the room (or ripped out the tether). The Command

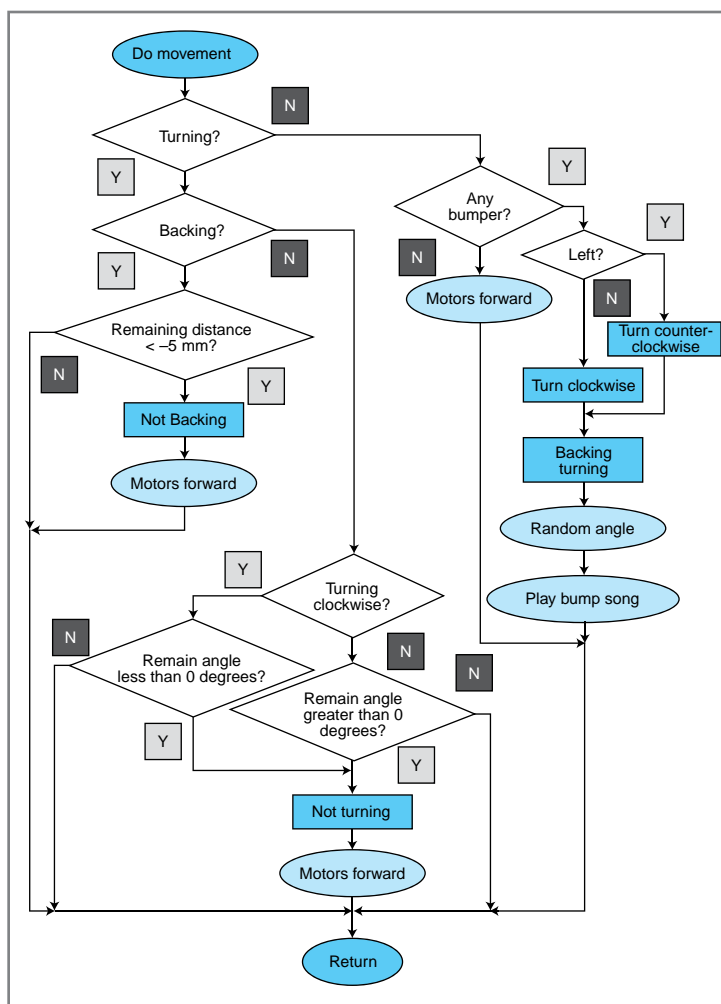


Figure 2—This is how the example DRIVE.C code handles the movement of Create. I based my assembly code on this flowchart.

| Command              | Opcode | Data bytes                                 |   |                                      |   |
|----------------------|--------|--|---|--------------------------------------|---|
|                      |        | 1  | 2   | 3                                    | 4, Etc.   |
| Start                | 128    |  |   |                                      |   |
| Baud                 | 129    | Baud code: (0–11)                          |   |                                      |   |
| Control              | 130    |  |   |                                      |   |
| Safe                 | 131    |  |   |                                      |   |
| Full                 | 132    |  |   |                                      |   |
| Spot                 | 134    |  |   |                                      |   |
| Cover                | 135    |  |   |                                      |   |
| Demo                 | 136    | Demos (–1 to 9)                            |   |                                      |   |
| Drive                | 137    | Velocity<br>(–500 to 500 mm/s)             | Radius<br>(–2,000 to 2,000 mm)            |                                      |   |
| Low-side drivers     | 138    | Output bits (0–7)                          |   |                                      |   |
| LEDs                 | 139    | LED Bits (0–10)                            | Power LED Color<br>(0–255)                | Power LED intensity<br>(0–255)       |   |
| Song                 | 140    | Song number<br>(0–15)                      | Song length<br>(1–16)                     | Note number 1<br>(31–27)             | Note duration 1<br>(0–255)<br>Note number 2, etc. |
| Play                 | 141    | Song number: (0–15)                        |   |                                      |   |
| Sensors              | 142    | Packet ID: (0–42)                          |   |                                      |   |
| Cover and dock       | 143    |  |   |                                      |   |
| PWM Low-side drivers | 144    | Low-side driver 2 duty cycle (0–128)       | Low-side driver 1 duty cycle (0–128)      | Low-side driver 0 duty cycle (0–128) |   |
| Drive direct         | 145    | Right wheel velocity<br>(–500 to 500 mm/s) | Left wheel velocity<br>(–500 to 500 mm/s) |                                      |   |
| Digital outputs      | 147    | Output bits<br>(0–7)                       |   |                                      |   |
| Stream               | 148    | Number of packets packet ID 1<br>(0–42)    | Packet ID 2, etc.                         |                                      |   |
| Query List           | 149    | Packet ID 1<br>(0–42)                      | Packet ID 2, etc.<br>Pause/Resume         |                                      |   |
| Stream               | 150    | Range:<br>0–1                              |   |                                      |   |
| Send IR              | 151    | Byte (0–255)                               |   |                                      |   |
| Script               | 152    | Script length: (1–100)                     | Command opcode 1                          | Command data byte 1                  | Command opcode 2, etc.                            |
| Play script          | 153    |  |   |                                      |   |
| Show script          | 154    |  |   |                                      |   |
| Wait time            | 155    | Time<br>(0–255 seconds/10)                 |   |                                      |   |
| Wait distance        | 156    | Distance<br>(–32,767 to 32,768 mm)         |   |                                      |   |
| Wait angle           | 157    | Angle<br>(–32,767° to 32,768°)             |   |                                      |   |
| Wait event           | 158    | Event ID<br>(1 to 20 and –1 to –20)        |   |                                      |   |

**Table 1**—The Create’s basic unit runs an operating system that enables you to manipulate behavior by controlling and monitoring its I/O devices through its serial port. The commands listed here can be sent by any device capable of serial communications.

Module has plenty of flash memory storage space to hold a useful control program.

Before I get into the programming of the Command Module, I need to mention that the Command Module passes the I/Os available on the 25-pin expansion connector to the user along with a bevy of extra I/O controlled directly from the microcontroller. These are accessed via four ePorts (DB9F). In addition to the extra I/Os, there is a RESET button, a User button, and a USB device port. The

USB port provides a programming tether to a PC. The RESET button jump-starts the Command Module. If a USB cable is plugged in, the microcontroller goes into Bootload mode to accept code via USB. If RESET is pushed without the USB cable plugged in, it will begin executing the downloaded code.

## COMMANDING THE SYSTEM

The Command Module’s CD installs a development

**Listing 1**—Each sensor has its own formatting code. Note that the first line of each routine has a `rem'd` statement that allows the routine to be skipped by removing the `;` (`rem`). Both of these routines use a conversion to decimal format on the 16-bit data in `r1` and `r0`.

```

;
DoDebug_SenDist:
;   jmp     DoDebug_SenAng
;   ldi     char2send, 'D'
;   call    byteTx
;   ldi     char2send, '='
;   call    byteTx
;   lds     r1,          SENSORS_OUT+SenDist1
;   lds     r0,          SENSORS_OUT+SenDist0
;   call    SendDecimal
;
DoDebug_SenAng:
;   jmp     DoDebug_SenChargeState
;   ldi     char2send, 'A'
;   call    byteTx
;   ldi     char2send, '='
;   call    byteTx
;   lds     r1,          SENSORS_OUT+SenAng1
;   lds     r0,          SENSORS_OUT+SenAng0
;   call    SendDecimal
;
DoDebug_SenChargeState:

```

system on your PC, which enables you to develop programs in C using the provided programmer's notepad IDE (editor), GNU GCC (compiler), and AVRdude (downloader). After installing this software, you can explore the three example programs on the CD. The first example, DRIVE, shows how to make the Create move around and react to an object when its bumper signals come in contact with it. The second example, INPUT, teaches you how to interact with the Create's I/O and the Command Module's extra I/O. The third example, LIGHT, introduces you to the process of adding external sensors. In this example, an added CDS light sensor (to the command module) is monitored while the Create wanders the area. When a significant increase in light is detected, the Create stops and plays a song.

This might be where we part company. As you probably know, I like getting down and dirty, so I'm going to finish up this column by discussing using assembly language with the Command Module, which I couldn't find much about on any of the Create-related sites or forums I visited. Although I think you get a great little development system here for the Create with the Command Module, there is a missing piece that is important to me when I develop something: debugging. A debugging

connector could have been added to the Command Module, which would have made in-circuit debugging simple. There are a few mentions of how to use the USB port for feedback, but there is conflicting information and there aren't any examples. I'm going to show you how to do this. Although I'll be working in assembler, this can be done in C as well.

### AVR Studio

I've used AVR Studio (IDE) in the past for development work with Atmel parts. It includes a simulator (which I use from time to time to check routines) and other tools for debugging (i.e., in-circuit emulation). Because I didn't have access to the debug port on the ATmega168 used in the Command Module (I'm sure they wouldn't want the user to accidentally wipeout the bootloader), I couldn't use this tool.

Using the C code as a guide, I went through the DRIVE example and came up with a flowchart. Figure 1 shows the initialization and idle loop waiting for a User button push. The movement routine in Figure 2 then takes over until a second push or any safety issues, such as a cliff sensor, are detected. Items in round bubbles are calls made to other routines. These can easily become a library of calls used over and over by your programs (just like a library in C).

A couple of the routines are worth mentioning. When a bumper is detected, a turn is requested with the direction based on the bumper so Create turns away from the collision. My random angle routine uses the 7 least significant bits of the Timer1 register counter (which increments rapidly with no prescaler). The constant 53 is added to this value (0–127) to allow angles between 53° and 180° (or –53° and –180° degrees when rotating clockwise). A drive command is then issued using a velocity of 200 mm/s with either radius of 0xFFFFmm (for clockwise rotation) or 0x0001mm (for counterclockwise rotation). The random angle value becomes the requested angle and is compared to the angle returned by Create to determine when the turn is completed.

The update routine requests a full complement of data to be returned by Create. Refer back to the Sensors command in Table 1. This command requires 1 data byte (0–42). There are in fact only 36 sensor values that you can ask for, so why isn't the data byte (0–35)? The first seven values (0–6) request groups of sensors to be sent. Value 0 requests the first 20 sensors, 1 requests the first 16, 2 requests sensors 11–14, 3 requests sensors 15–20, 4 requests sensors 21–28, 5 requests sensors 29–36, and 6 requests all 36 sensors. This makes using this command quick. Because the commands going to and sensor data being returned from Create happen asynchronously (i.e., your program may execute numerous instructions while the sensor data is being received), you must be cautious about when you rely on new data. For this reason, once all the sensor data has been received, it is copied into a working bank of registers for you to access. This approach assures each time you access sensor data it is complete.

There is sensor data received (e.g., distance and angle) that must be used appropriately each time it is received. These items indicate a change since the last request and are cleared internally (within Create's OS) when sent. If you use the drive command to rotate counterclockwise, you must

continually add the sensor angle values to determine when you've rotated the proper amount. You must then use the drive command to halt the motors (velocity = 0). Distance works the same way. While the value of every sensor available is not used in this program, I ask for it all any way, just as an exercise. It also makes the next section more interesting, as you will see shortly.

## DEBUGGING

You don't have a way of getting inside the microcontroller and single-stepping execution or looking at register values, but you can get the next best thing. The single UART in the ATmega168 is actually directed by external circuitry toward either the USB port for bootloading your program or Create's serial port, via the 25-pin expansion connector, to respond to your program's command requests. The control that does this switching is an output bit on the ATmega168. You have access to this, which means from your program you can flip back and forth between destinations. Because the microcontroller is set up to go into bootload mode whenever a USB cable is plugged into the Command Module's USB port, swapping destinations is impossible unless you know the secret. Note that holding down the User button while resetting the microcontroller can deactivate this function.

For debugging purposes, I can write a routine to send the sensor data out of the USB port each time the sensor data is copied into the working registers. Note that even though you can check the UART to see if any characters remain in the transmitter, you should add a delay of at least one character before switching the serial output destination to ensure that you are not interfering with a transmission in process.

To view the debugging data, you can use a terminal emulator like RealTerm. Because you are now anchored again with tether, you will want to place the Create up on a support to keep its drive wheels off the ground. You will also need to comment out the safety checks within

the program so the Do Movement loop isn't exited when one of the cliff sensors is triggered. I use conditional assembly with a Debug mode that eliminates certain code. I started by just sending the data bytes as received. While RealTerm enables you to see the data in various formats such as ASCII, binary, or HEX, there are a few problems with this, as you might expect. Not all of the data makes sense in any one format. And some sensor values are multibyte representations of a word value. It makes sense to format each sensor value appropriately and send the value as ASCII characters. For bytes used as bit flags, binary might be appropriate. For signed word values, a + or - sign would be appropriate, and so on. You can get as fancy as you wish. A name or an alias that helps to identify it would also be useful. You don't have to limit the data to just the Create's sensor values. You can display any variables your program is calculating.

Be careful what you choose to display. Too much data can be just as bad as too little. Your screen can fill quickly. Logging to a file will be helpful in some instances. The idea is to help you narrow down where to look for a bug in your code. When the robot backs up and rotates forever, it might not be so obvious where to start looking in your code. I filled

my debug output routine with every possible piece of data to export. I added a `jmp` instruction at the beginning of each section of code for outputting that particular piece of data (see Listing 1).

I can quickly REM out the `jmp` instruction of any sections I want to include in the debug display. Photo 2 shows power-related data I monitored from the Create. Note that the number of variables you choose to display (total characters sent) will impact the execution loop speed of the program.

In AVR Studio, I created a project with three files. The first file, `m168def.inc`, is the register/bit definitions file for the ATmega168. The second file, `CreateCmdDef.inc`, is used to hold the definitions for all of the items directly relating to the Create. I also placed the callable routines I made to support the Create's commands and my Debug code. This kept the ugliness out of the main file. The last file, `iRobotCreateTest1.asm`, is the main file. This is the control program. It is the only file that must be written for a new project. Assuming all of the commands have working code in `CreateCmdDef.inc`, the main code has includes that link the first two files.

One of the nice things about using AVR Studio is that it contains a simulator. I often use the simulator to calculate routine timing using the integrated

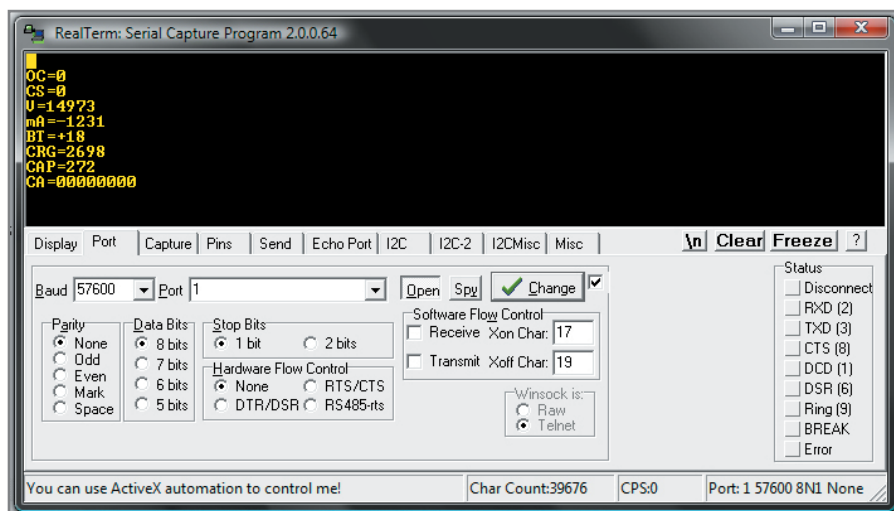


Photo 2—RealTerm is displaying some of the internal register values that have to do with power. OC = 0 (no over-currents), CS = 0 (not charging), V = 14,973 (battery = 14.973 V), mA = -1,231 (batteries supplying 1.231 A), BT = 18 (battery temperature = 18°C), CRG = 2,698 (2.698 A/h remain), CAP = 272 (battery capacity is down by 0.272 A), and CA = 00000000 (no source).



stopwatch. I even use it to learn a routine I've copied from somewhere else. After assembling the code, I use AVR-Dude (installed with the Create CD) to download my .HEX file to the Create through the command module's USB port. If I have the Create up on blocks, I can execute from here by holding down the User button while pressing the Reset button. This can be done when the Create is off, because the Command Module has its own power switch. When the Command Module begins execution, it can check the status of the Create (through the 25-pin expansion connector) and turn it on, if necessary!

In this state, you can run it all while the wheel spins without floor contact, pressing buttons and bumpers and watching for proper responses. Of course, you'll want to pull out the tether and try it on the floor even if it is misbehaving. But use caution. If you aren't paying attention to safety issues, the Create may try to run over the cat if the bumper routine doesn't work. Or it might continue backing up if you aren't paying attention to distance because there "are no bumpers in the rear."

Now I need to think about what kind of external sensors I might want to add to support a mapping project. This should be interesting.

## ARE YOU CREATIVE?

If you want to check out some of the projects others are working on, visit the Create projects link in the Resources section. You will find a bevy of third-party add-ons that were developed for the Create. Refer to <http://forums.irobot.com/irobot/home> and [www.avrfreaks.net](http://www.avrfreaks.net) for discussions about iRobot and Atmel-related things.

If you want to work with an affordable robotic platform without having to deal with building and testing hardware from scratch, this is an ideal vehicle. If you want to begin with a proven design and add your own sensors, this is an ideal platform. While the top-of-the-line Create with more stuff than I've discussed here costs \$300, the basic unit begins at approximately \$100. It can really help

jump-start your robotics research, product development, or technical education!

Next month, I will explain how to

replace the Command Module with a microcontroller. You are well on your way to developing your first application. ☒

*Jeff Bachiochi (pronounced BAH-key-AH-key) has been writing for Circuit Cellar since 1988. His background includes product design and manufacturing. You can reach him at [jeff.bachiochi@imaginethatnow.com](mailto:jeff.bachiochi@imaginethatnow.com) or at [www.imaginethatnow.com](http://www.imaginethatnow.com).*

## RESOURCES

iRobot Corp., "iRobot Create Open Interface Specification," 2006, [www.irobot.com/filelibrary/pdfs/hrd/create/Create%20Open%20Interface\\_v2.pdf](http://www.irobot.com/filelibrary/pdfs/hrd/create/Create%20Open%20Interface_v2.pdf).

———, "Command Module Owner's Manual," 2007, [www.irobot.com/filelibrary/pdfs/hrd/create/Command%20Module%20Manual\\_v2.pdf](http://www.irobot.com/filelibrary/pdfs/hrd/create/Command%20Module%20Manual_v2.pdf).

———, Create projects, [http://store.irobot.com/family/index.jsp?categoryId=2591511&ab=CMS\\_IRBT\\_CreateSuperCat\\_LearnMore\\_110408](http://store.irobot.com/family/index.jsp?categoryId=2591511&ab=CMS_IRBT_CreateSuperCat_LearnMore_110408).

## SOURCES

### ATmega168 Microcontroller

Atmel Corp. | [www.atmel.com](http://www.atmel.com)

### Create programmable robot

iRobot | [www.irobot.com](http://www.irobot.com)

### RealTerm Terminal emulation software

RealTerm | <http://realterm.sourceforge.net>

**Windows CE Touch Controller**  
**CUWIN™**

CUWIN4300K ▶ **\$499** / Qty.1

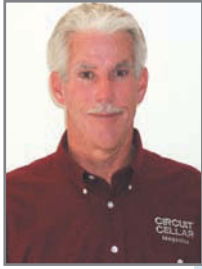
- 7" / 10.2" wide color TFT display
- 800 x 480 resolution, 260K colors
- SD card & Ethernet support
- RS232 x 2 / RS485 x 1 or RS232 x 3
- Mono Speaker and Stereo Jack
- Real time clock (Battery backup)
- Visual Basic and Visual C++ support
- USB I/F (ActiveSync)
- ARM9 32bit 266MHz processor
- Keyboard or Mouse support

CUWIN3200 ▶ **\$399** / Qty.1

CUWIN3200, CUWIN3500, CUWIN4300K, CUWIN4300S

- \* CUWIN3200: 7" Bezel type case
- \* CUWIN3500: 7" Cover Case (Front waterproof)
- \* CUWIN4300K: 10.2" Black Bezel type case
- \* CUWIN4300S: 10.2" Silver Bezel type case

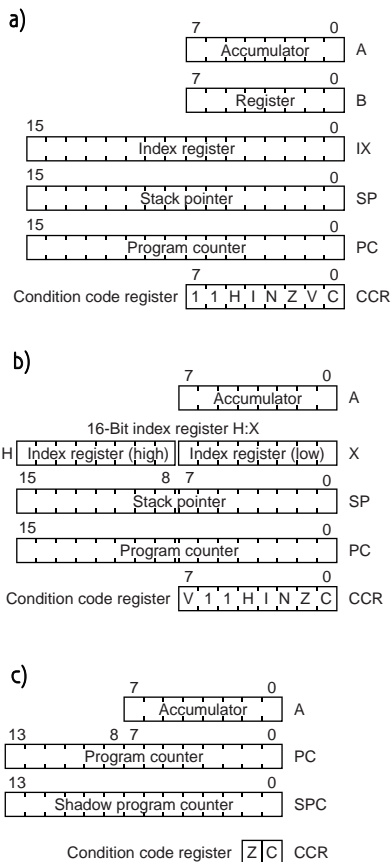
**COMFILE TECHNOLOGY** [www.cubloc.com](http://www.cubloc.com)  
Toll-Free: 1.888.928.2562



## A Really Simple Plan

### The “8-Bits” Saga Continues

Thirty-two-bit microcontrollers are becoming more popular, but 8-bit chips still have their place in the market. Tom says major companies will be finding uses for the simple and inexpensive chips for years to come.



**Figure 1**—Judging by the similarity between the original Freescale Semiconductor MC6800 (a) and the modern MC9508 (b) you wouldn't guess 30 years separates them. By contrast, the new Freescale MC9R508 (c) is noticeably different.

“8-bits is dead.”

That's what a micro marketing expert told me. What's funny is that he said that 30 years, and an untold billions of 8-bit chips, ago. But hold on. The story gets even better. A few years later, I bumped into this fellow again and learned he had moved on to a new job at a different chip company. Things were going quite well and he was making lots of money—selling 8-bit micros!

You've heard me tell that story before, and no doubt I will tell it again. Yes, 32-bit MCUs are on the march and growth is explosive. And it's true there's an ever-shifting gray area between “high-end” 8-/16-bit chips and “low-end” 32-bit chips where it makes perfect sense for the latter to replace the former. But the problem with the 8-bit eulogies is that they're based on the premise that silicon is a zero-sum game. If 32-bit MCUs go up, that means 8-/16-bit MCUs must go down, right? Zero sum may be the way it is for some businesses, but not silicon. As long as “Moore for Less” is the name of the game, everybody wins.

And don't forget to look at the big picture. How many 8-/16-bit MCUs touch your life? They're in virtually every electronic gadget. Now consider developing societies on a global scale. Those folks are going to want lots of gadgets too.

There's one more bit of evidence that's testimony to the staying power of our little friends. The fact is all the major players in the 8-/16-bit MCU biz continue to roll out new parts. That's

**NEW!**




**Robot Kits**  
Line followers  
Robot arms  
Hexapods  
Chassis


**3pi Robot**  
\$99.95

High-performance, C-programmable, ATmega168-based robot (with Arduino support)!

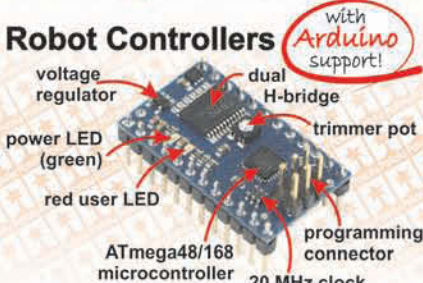
**Mechanical Components**  
Motors, servos  
Wheels, ball casters



**Motion Control**  
Motor controllers  
Servo controllers



**Robot Controllers** *with Arduino support!*



voltage regulator  
power LED (green)  
red user LED  
ATmega48/168 microcontroller  
20 MHz clock  
dual H-bridge  
trimmer pot  
programming connector

**Solder Paste Stencils**  
Use our low-cost solder paste stencils to quickly assemble your surface-mount designs.

**From \$25**



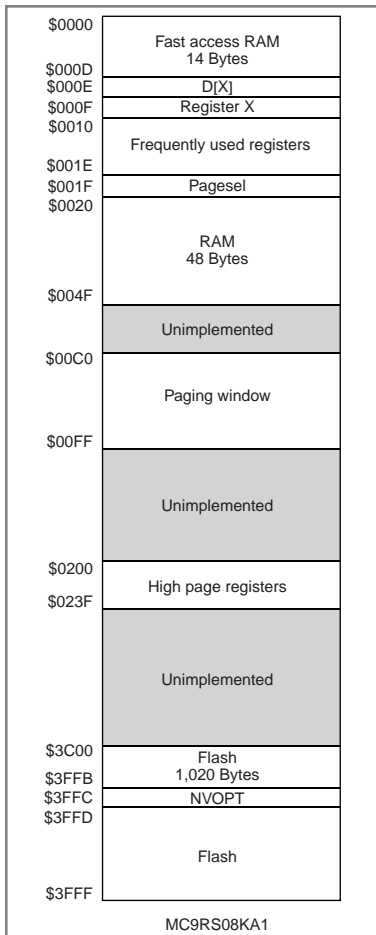
**Custom Laser Cutting**  
**From \$25**



Cut your own custom chassis, front panels, and more!

**1-877-7-POLOLU**  
**www.pololu.com**

6000 S. Eastern Ave. 12D, Las Vegas, NV 89119



**Figure 2**—The 'RS08 memory map reveals details about the architecture. The direct address space is only 256 bytes (\$0–FF), so higher addresses are accessed via a 64-byte “paging window” (\$C0–FF). New TINY and SHORT addressing modes access low memory including fast access RAM, the emulated X index register, and the most frequently used I/O and control registers.

all the way to the 1970s when Freescale-then-Motorola introduced their first micro, the MC6800. The architecture of the MC6800 was blessedly simple, comprising little more than two 8-bit registers (A and B), an index register (X), and the usual stack pointer (SP), program counter (PC), and condition code (CC) registers (see Figure 1a).

Over the following decades (i.e., 1980s and 1990s), Motorola introduced a number of variations on the MC6800 theme. Along the way, there was a fork in the road with classic favorites like the MC68HC11 taking the high road and the down-sized MC68HC05 taking the low.

While the MC68HC11 and MC68HC05 remain mainstays in the catalog, the modern era (roughly corresponding to the Freescale spinout) has been centered on the core known as the 'S08. What's remarkable after some 30 years is how true the 'S08 remains to the original (see Figure 1b). The menu has grown to include hundreds (heck, maybe thousands at this point) of distinct parts crafted from a few basic ingredients.

Given that history, the new 'RS08 stands out by virtue of the fact that it's noticeably different, something immediately apparent with a glance at the programmers model (see Figure 1c). The

hardly symptomatic of a “dead” market.

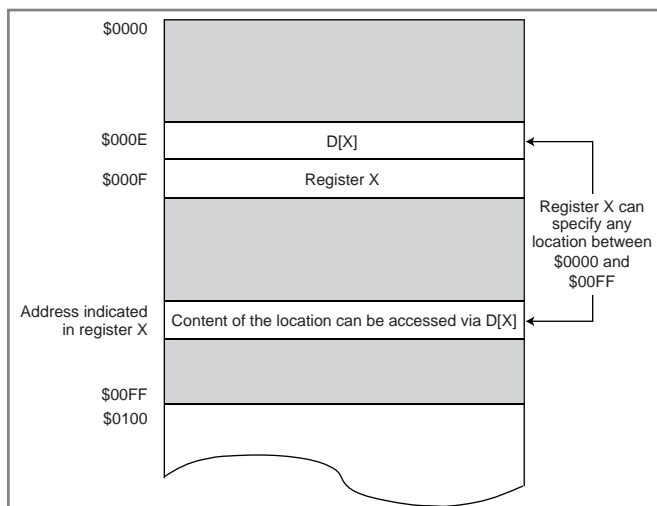
Freescale Semiconductor is no exception. They've been one of the top players in the MCU market since the very beginning. And judging by the fact that they've introduced a brand new 8-bit processor core, the 'RS08, they plan on keeping it that way.

## FORWARD TO THE PAST

To get a better understanding of what the new 'RS08 core is all about, let's start with a bit of a history lesson.

Like many other popular MCUs, the 'RS08 has roots that go way back,

**Figure 3**—The X index register hasn't gone missing. Instead, it's emulated using a pair of memory locations (X and D[X]) in the TINY address space.



MC6800 and its descendants were lean and mean, but the 'RS08 is positively anorexic. You've all seen the movie *Back to the Future*, right? Well the 'RS08 is kind of a "For ward to the Past" scenario, with Freescale introducing a core today that's architecturally simpler than the circa-1970s original.

After a glance at the programmers model, the first question that arises is how in the heck can the 'RS08 get anything done? Yeah, there's an accumulator (A), but just how are you supposed to get data into or out of it with the disappearance of the index (X) register? The answer is a combination of direct addressing (i.e., memory address included in the instruction), paging, and an X index register in drag.

Here's how it works. The entire address space for the 'RS08 is 16 KB, comprising 256 pages of 64 bytes (see Figure 2). But the direct addressing range tops out at just 256 bytes (i.e., 8-bit address) and the only instructions that can specify a longer address (i.e., the full 14 bits) are JMP and JSR. To provide data access to the entire 16 KB, the top 64 bytes of the direct address space (i.e., page 3, direct addresses \$C0-FF) are used as a window to any other page as specified with a page select (PAGESEL) register.

Index register X didn't really disappear, it just morphed into a memory-mapped emulation. As you can see in Figure 3, location \$000F doubles as the X register (8 bits as in the MC68HC05) and location \$000E in turn acts as a holder for the data addressed. Indeed, this scheme can partially emulate MC68HC05-indexed addressing (albeit more slowly) to the point that the assembler accepts some traditional X register mnemonics as "psuedoinstructions" and automatically converts them to 'RS08 equivalents (i.e., access via locations \$000E and \$000F).

Now let's focus a little on the "R," as in "reduced," aspects of the 'RS08. While the maximum direct address is 8 bits, there are new "SHORT" (5-bit) and "TINY" (4-bit) direct-addressing modes that work with certain instructions. Carefully craft your data layout to put the most frequently accessed data in the TINY space (first 16 bytes). Meanwhile, the most frequently used I/O and

| Instruction | Addresses |
|-------------|-----------|
| INC         | \$00-\$0F |
| DEC         | \$00-\$0F |
| ADD         | \$00-\$0F |
| SUB         | \$00-\$0F |
| CLR         | \$00-\$1F |
| LDA         | \$00-\$1F |
| STA         | \$00-\$1F |

**Table 1**—Thanks to new SHORT and TINY addressing modes, the lower 32 bytes of the address space are uniquely accessible with single-byte versions of the most common instructions.

control registers (including the aforementioned PAGESEL register) are mapped into the remaining SHORT space (bytes 17–32). In return for taking full advantage of the scheme, you get the speed and code density advantages of single-byte instructions (see Table 1).

SP is gone, and as you might imagine, so is the stack. Instead, there's just a "Shadow PC" register (SPC) that acts as a single-level stack for interrupts and subroutine calls. In a pinch (e.g., you need nested subroutine calls), instructions to move data between the accumulator and the high and low bytes of SPC give you hooks to mimic a hardware stack (i.e., PUSH and POP macros).

Without getting sidetracked by the

now rather tired RISC-versus-CISC debate, it's pretty clear that the 'RS08 is still a CISC because instructions can operate directly on memory. It's just a "Complex Instruction Set Computer" that happens to have a "Reduced Instruction Set." Given what you've seen so far, it is no surprise that entire categories of traditional (i.e., 'S08) instructions (such as those having to do with SP and X) are vaporized. Further, even the instructions that remain have fewer variants (e.g., no 16-bit direct address mode, only zero-offset indexing). We're not talking about a minor weight loss. Considering both the number of basic instructions, as well as all possible opcodes (i.e., combinations of instruction and addressing mode), the 'RS08 instruction repertoire is slashed to well under half the 'S08. According to Freescale, the 'RS08 core is fully 30% smaller than the 'S08.

## INTERRUPTS DISABLED

As you might expect, the absence of a "real" (i.e., hardware stack) poses a challenge for dealing with interrupts. The 'RS08 has a straightforward approach to the problem. If interrupts are a hassle, ditch them altogether. You heard me

**Listing 1**—The way the 'RS08 handles interrupts is simple. It doesn't. Instead, an interrupt simply sets a bit in a flag register and wakes up the processor. It's up to software to figure out what's going on and what to do about it.

```

InfLoop:
    sta SRS ;Bump COP
    wait

P1:  brset SIPI_MTIM,SIPI_MTIM_ISR ;5 bus cycles
P2:  brset SIPI_ACMP,SIPI_ACMP_ISR ;5 bus cycles
P3:  brset SIPI_KBI,SIPI_KBI_ISR ;5 bus cycles
P4:  brset SIPI_RTI,SIPI_RTI_ISR ;5 bus cycles
P5:  brset SIPI_LVD,SIPI_LVD_ISR ;5 bus cycles
Bra  InfLoop

MTIM_ISR:
;... <ISR coding> ...
    bra InfLoop
ACMP_ISR:
;... <ISR coding> ...
    bra InfLoop
KBI_ISR:
;... <ISR coding> ...
    bra InfLoop
RTI_ISR:
;... <ISR coding> ...
    bra InfLoop
LVD_ISR:
;... <ISR coding> ...
    bra InfLoop

```

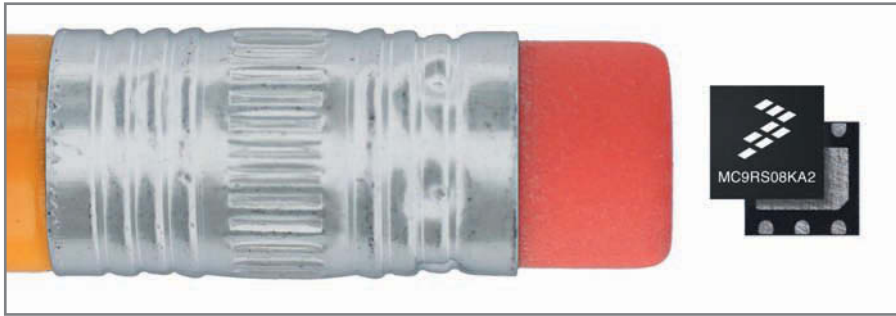


Photo 1—The 'RS08 six-pin DFN package is so small it will erase your board space concerns.

right. The 'RS08 simply doesn't do interrupts, at least in the conventional sense.

There is a measure of attention-getting capability from the usual sources (i.e., internal peripherals and external pins). However, unlike a conventional interrupt scheme, a source request doesn't affect a running program. Rather, it does only two things. First, it sets a flag bit in a register, signifying the source. Second, it wakes up the processor if it's sleeping.

Needless to say, without true interrupts, there's certainly no need for the fancy add-ons such as vector, priority, and nesting schemes. If you've already thrown out the baby, you may as well go ahead and throw out the bathwater. Instead, what passes for interrupt handling boils down to a combination of waking up (periodically and/or in response to an internal or external source event) and polling the event flag bits to determine the source.

Listing 1 shows a likely application software scenario. The program is an infinite loop (InfLoop) that simply waits (WAIT instruction) in low-power mode until awoken by an internal or external event. Then the program proceeds (P1) to scroll through the list of possible sources, checking the event flags and branching (via the BRSET branch-if-bit-set instruction) to the handler for the event that generated the wake-up call. After the handler does its thing, it returns to the beginning of the loop and goes back to sleep to wait for another call to action.

The polling order has two ramifications. First, it's a way to brute force the issue of priority because events higher up the list will be polled and, if pending, handled before those further down the list. It also means latency between the

occurrence of the source event and entering the associated handler varies depending on the event's polling position. In the example shown here, at top speed (10-MHz bus cycle), the latency grows from about 1 to 3  $\mu$ s between the first (MTIM timer) and last (LVD low-voltage detect) events in the polling order. Actual latency may be stretched even further depending on device-specific wake-up delays, such as the time it takes for an on-chip voltage regulator or clock generator to power-up and stabilize.

Is it conceivable to craft clever software to mimic the fancy features of a hardware interrupt controller? Sure, a pro could probably even cobble together some kind of midget RTOS. Is it a good idea? Probably not. If in doubt, I'd say you're better off going with a bigger-ticket part (e.g., 'S08) that has a real stack and interrupts. On the other hand, there are many, many applications that can easily get by with the 'RS08's simple scheme. When every penny counts, why pay for stuff you don't need?

## MINI-MEM

The initial parts in the 'RS08 line-up come with just 1 KB (MC9RS08KA1) or 2 KB (MC9RS08KA2) of flash memory. Follow-on parts have more, as you'll see shortly, but remember the architecture itself accommodates only a 16-KB address space. Notwithstanding the fact that 'RS08 code density is enhanced by the 1-byte TINY and SHORT instructions, bloatware aspirations will surely be better served by a higher-end MCU.

Many flash memory MCUs feature in-application programming capability that enables code updates in the field and also allows portions of the flash memory to be used as a "pseudo-EEPROM" for data storage. By contrast, it's

important to note that the 'RS08 flash memory is really intended to be programmed just once as it goes out of your factory door.

There are a number of datasheet specs that make this conclusion obvious. Unlike flash memory parts that can typically be written 10,000 or more times, the 'RS08 flash memory write endurance spec is just 1,000 cycles. In their frenzy to jettison ballast, the 'RS08 designers also tossed the charge pump needed to generate the higher flash memory programming voltage (12 V), so you're responsible for providing it externally. And while the flash memory is organized and programmed on a 64-byte row basis, there's only an all-or-nothing mass-erase capability. That means using the flash memory for multiple in-application writes would require a "rolling" update scheme that writes only once to a given location. For example, if you have 2 bytes of nonvolatile data you'd like to update from time to time, allocating a 64-byte row of the flash memory would allow for 32 updates.

Nevertheless, anything is possible and Freescale even has an application note describing such a "pseudo-EEPROM" scheme.<sup>[1]</sup> It relies on moving a small flash-programming routine from flash memory into RAM for execution because flash memory access (e.g., instruction fetch) is not allowed during programming. The technique is rather scary and fraught with peril because once flash memory programming is underway, any hiccup ("Oops, I forgot to disable the watchdog timer") could leave the part brain-dead. And while it's nice that code can be executed from RAM, don't get too excited because the MC9RS08KA1 and MC9RS08KA2 give you only 48 bytes to play with.

## SMART DUST

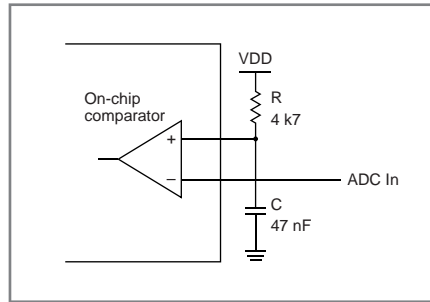
Sticking with the "small is beautiful" theme, the MC9RS08KA1 and MC9RS08KA2 come with just six or eight pins. The largest (if you can call roughly 6 mm  $\times$  10 mm large) of these is a vintage eight-pin DIP, which is useful for prototyping or cheap (e.g., single-sided through-hole) PCBs. You can cut the DIP's already small footprint in half by going surface mount with the optional eight-pin leaded SOIC package at just

Trinity College  
presents:

# 16th Annual Fire Fighting Home Robot Contest

Join robot enthusiasts  
of all ages as they  
gather and compete  
in one of five divisions  
at Trinity College  
in Hartford, CT.  
See how an exciting  
Robotics Symposium  
and Olympiad help  
spark the flame of  
robot innovation!

Saturday & Sunday  
**April 4 & 5, 2009**  
[www.trincoll.edu  
/events/robot](http://www.trincoll.edu/events/robot)



**Figure 4**—The MC9RS08KA analog comparator can mimic a conventional ADC. Software discharges the RC and then the MTIM timer measures how long it takes to recharge to the level of the psuedo-ADC input.

5 mm × 6 mm.

If you're willing to give up a couple of pins, the six-pin (actually no pins, just pads) dual flat nonleaded (DFN) package represents a new low in board space. This Chihuahua is amazingly small at just 3 mm on a side and 1 mm thick (see [Photo 1](#)). Consider that you can cram half a dozen of the DFN parts in the footprint of the "large" eight-pin DIP.

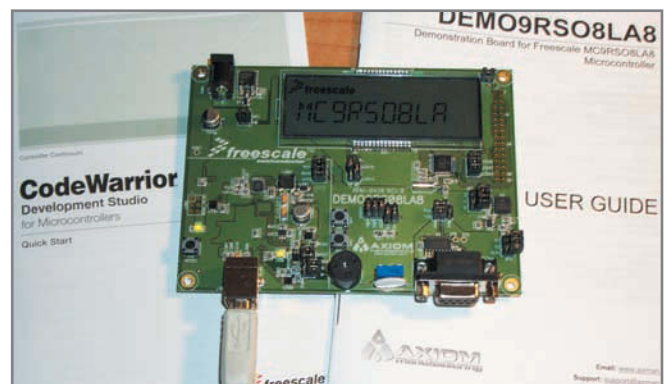
Needless to say, with so few pins, there's no need for a bunch of fancy I/O. We're talking just the basics, split between general system functions and pin-oriented peripherals. The list starts with an on-chip regulator so the 'RS08 will run on any voltage you care to provide between 1.8 and 5.5 V. There's also a built-in clock system based on an on-chip 32-kHz oscillator and a bypassable frequency-locked loop. The clock is trimmable to achieve 2% accuracy across the entire voltage and temperature (−40° to 85°C) range. Given the cost-cutting bias and dearth of pins, I'm not surprised there's no option for external clocking. For robustness, there's a watchdog timer, low-voltage detection, and a multisource RESET controller that accommodates the usual suspects such as power-up, power fail, watchdog timeout, and optionally even a pin. The RESET controller also keeps the street

safe for programmers by busting code that's up to no good (e.g., illegal opcode, invalid address).

Because no one has figured out how to get rid of power and ground, the built-in I/O has just four (DFN package) or six (SOIC, DIP) pins to play with. Naturally, any or all can be used as parallel I/O with pull-up and pull-down options, as well as slew-rate control. Any or all can also serve as inputs to a keyboard interrupt module (KBI). Recalling that the 'RS08 really doesn't have "interrupts" per se, the KBI's most likely use is to catch edges (rising, falling, either) and otherwise serve as a means to wake-up the processor from a low-power state.

Similarly, there's an analog comparator using up to three pins that can compare two external voltages to each other or to an internal reference voltage. You can choose to trigger on the rising, falling, or either edge of a comparator transition, and the comparator output can be routed to a pin. Freescale application note 3266 describes how to use the comparator to implement a poor man's ADC using the familiar R/C charge timing scheme (see [Figure 4](#)).<sup>[2]</sup>

The ADC emulation also uses the final piece of the I/O pie, the MTIM timer module. It's a simple unit with an 8-bit prescaler (1, 2, 4...256 divide ratio) fronting an 8-bit counter. Various internal clock sources can be selected as the counter input or a pin can be devoted to the cause. Note that the MTIM module is not capable of waking up the processor from low-power mode. Fortunately, you can



**Photo 2**—Freescale's introduction of the 'RS08 "L" series, and the evaluation kit that goes with it, proves that blue-collar technology (i.e., 8-bit MCUs and segment LCDs) lives on.

configure the processor's own clock module to generate a periodic (e.g., 1 ms) wake-up call.

Let's not overlook the small matter of debug. The 'RS08 background debug mode (BDM) provides the basic commands (e.g., read/write registers and memory, breakpoints, single-step) to get a handle on things. It's like a simple ROM monitor, but it is less intrusive (e.g., memory can be accessed without stopping the user program) and consumes no resources (e.g., flash memory or RAM) beyond the single pin (BKGD) used for host communication and flash memory programming.

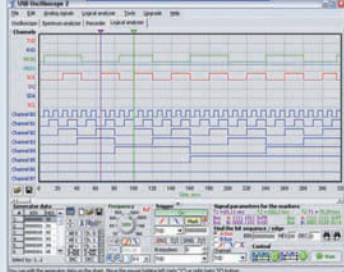
## MARKET SEGMENT

With the MC9RS08KA1 and MC9RS08KA2 staking their claim at the entry level, Freescale recently announced two high-integration 'RS08s, the 'LE4 and the 'LA8. In terms of the processor core itself, everything you've read so far applies, except there's more memory (4 or 8 KB of flash memory and 256 bytes of RAM) and larger 28- and 48-pin packages.

The real difference, and reason for the extra pins, is a significant boost in the on-chip I/O capability with MCU mainstays such as UART, SPI, fancier timers, and a multichannel 10-bit ADC. The final I/O add-on, and *raison d'être* for the 'L parts, is a built-in segment LCD controller.

Now I'm quite sure there's an LCD marketing expert somewhere that has pronounced "Segment LCDs are dead." Yes, it's true that bitmap graphics LCDs (e.g., 1/4 VGA) are all the rage and don't require any upfront NRE for custom glass, a big advantage for low-volume applications. But don't think the success of bitmap displays sounds the death knell for segment LCDs. They are still the best solution for high-volume apps where requirements for low unit cost and long battery life prevail. And that's not to say segment LCDs can't do eye candy. Witness some of the cool things being done in automotive dashboard displays, including free-form (i.e., nonrectangular) panels. Take a look around and you'll see segment LCDs are still a big deal

USB Oscilloscope for \$169.50  
Logic and Spectrum Analyzers, Generator.  
[www.HobbyLab.us](http://www.HobbyLab.us)

Celebrating Our 10<sup>th</sup> Year Anniversary

You Can't Have  
**Electronics Without The  
TOOLS to Build Them!**

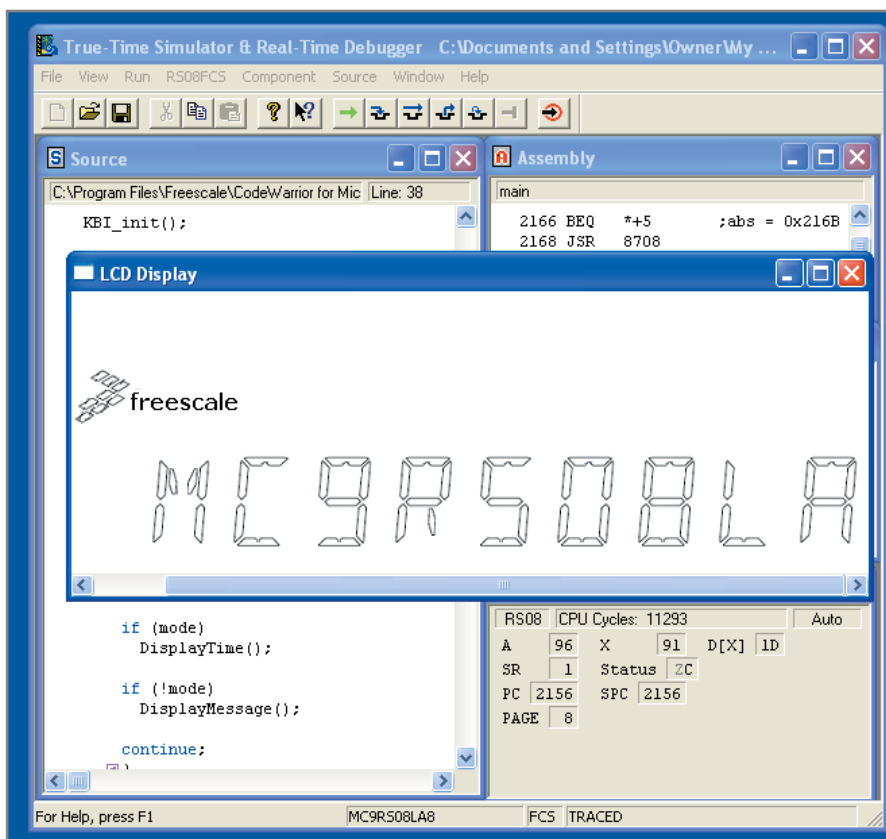
20% Sale On Wide  
Selection Of  
**Tools & Tool  
Sets!!**



Solder/ Desolder  
Tweezers  
Magnifiers  
Drills  
Screwdrivers  
Anti-Static Mats  
Crimping  
& More...

DesignNotes.com  
What Your Electronic Hobby Stores Used To Be

1-800-957-6867 [www.DesignNotes.com](http://www.DesignNotes.com)



True-Time Simulator & Real-Time Debugger

Source: C:\Program Files\Freescale\CodeWarrior for Mic (Line: 38)

```
KBI_init();
```

Assembly: main

```
2166 BEQ *+5 ;abs = 0x216B
2168 JSR 8708
```

LCD Display

freescale

MC9RS08LA

```
if (mode)
  DisplayTime();

if (!mode)
  DisplayMessage();


continue;
```

RS08 CPU Cycles: 11293

|      |      |        |      |      |    |
|------|------|--------|------|------|----|
| A    | 96   | X      | 91   | D[X] | ID |
| SR   | 1    | Status | ZC   |      |    |
| PC   | 2156 | SPC    | 2156 |      |    |
| PAGE | 8    |        |      |      |    |

For Help, press F1 MC9RS08LA8 FCS TRACED

**Photo 3**—The CodeWarrior 'RS08 toolchain that comes with the evaluation kit includes a full-featured simulator that encompasses software, hardware (MCU pins and peripherals), and even external I/O devices like the LCD.



CIRCUIT CELLAR  
FOR COMPUTER APPLICATIONS

Make sure you're signed up to receive Circuit Cellar's monthly electronic newsletter. *News Notes* will keep you up to date on Circuit Cellar happenings. Stay in the loop!

Register now. It's fast. It's free.  
[www.circuitcellar.com/newsletter/](http://www.circuitcellar.com/newsletter/)

in applications such as appliances, thermostats, calculators, glucose meters, not-so-smart phones, and hand-held test equipment (e.g., digital VOM). The list goes on. And just as with 8-bit MCUs in general, if segment LCDs are dead, why is it that heavyweights like Freescale and all of the other major players (Atmel, Microchip Technology, Texas Instruments, and more) keep introducing new segment LCD parts?

For the simplest (i.e., fewest segment) apps, you may be able to get away with a software bit-banging "static" approach that devotes a pin to each segment. But the number of segments (and thus pins) required can balloon fast. For example, a single alphanumeric digit requires up to 16 segments.

Enter the "multiplexed" alternative, which like a core memory of yore, uses an XY (frontplane and backplane in LCD-speak) electrode scheme, where a single pin controls multiple segments in a time division fashion, slashing the number of pins required. For instance, with 24 pins, a static controller can handle 24 segments. By switching to a multiplexed scheme with four back planes and 20 front planes, the same 24 pins can accommodate 80 segments.

The approach ups the ante in terms of processing and electronics complexity. LCD glass can be finicky with its demands for phase-shifting AC and carefully controlled voltage levels to avoid creating nearby "ghost" segments. High-segment-count multiplexed LCD control is one of those duties where a little hardware can go a long way to make life easier for designers.

Featuring 4:1 and 8:1 mux options, the 28-pin MC9RS08LE and 44-pin MC9RS08LA can handle up to 112 and 168 segments, respectively. Embellishments include a pin-mapping feature that facilitates PCB and connector layout to reduce EMI and crosstalk. There are also a variety of power options that accommodate connection with different flavors of glass (e.g., 3 V, 5 V, ST, and STN). Another nice touch is a blink attribute that handles that annoying chore (e.g., the blinking colon on a digital clock) without bothering the processor.

Freescale makes kicking the tires easy for you, and your budget, with the MC9RS08LE and MC9RS08LA evaluation kits that are a real bargain (see [Photo 2](#)). For just \$59, you get a board with an LCD plus the usual trimmings (buttons, LEDs, a buzzer, and more). And the boards have the USB debugger interface built-in, so you don't need a separate pod. The kits also come with the venerable CodeWarrior suite and, because the code size restriction for the evaluation version is 32 KB (i.e., larger than the flash memory on any 'RS08), you're essentially getting a proven and production-worthy 'RS08 toolchain for free.

Just before going to print, one of the MC9RS08LA kits appeared on my doorstep. I didn't have time to do more than install the tools, fire it up, and poke around a bit. Once I jumped through the obligatory install hoops (install CodeWarrior, install the service pack, install USB drivers, download the demo project from the web, and configure CodeWarrior/project options), everything I tried (compile, download, debug, and more) worked without a hitch. One feature that stood out was that the toolchain has a simulator that goes beyond just opcodes to include "full-chip" (i.e., pins) and even a measure of "system" simulation capability (see [Photo 3](#)).

While studying the MC9RS08LA board, I made an interesting discovery. Look closely at the right edge and you'll see a Freescale three-axis MEMS accelerometer. It is shown on the schematic, but otherwise isn't mentioned anywhere in the documentation.

*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](mailto:tom.cantrell@circuitcellar.com).*

It's all the more a bargain to get two evaluation kits for the price of one.

## KEEP IT SIMPLER

There is no doubt that some folks think all of this meat-and-potato technology is boring. But not me. I see the cost savings of these under-a-buck MCUs (well under for the smallest parts) passed on as a global "Tech Tax" cut for the masses. I see the milliamps-across-megaunits power savings as a green tsunami sweeping away a zillion batteries that would otherwise consume resources only to end up in a landfill. Dare I say I think this stuff is sexy and sure to spawn innovative applications. I applaud Freescale (and for that matter, all the other MCU suppliers hopping on the keep-it-real bandwagon) for standing up to the "experts" and naysayers who are enamored only with prima-donna technology.

My only advice to Freescale and the others is remember to keep it simple. And then remember you're supposed to remember to keep it simple. There are already plenty of feature-creeped, 8-bit MCUs to choose from. Don't get me wrong, the higher-end parts are wonderful too and I'm not necessarily against reinventing the wheel. But like putting fat tires and tail fins on a go-kart, tarding up a mini-me MCU isn't the way to do it.

I apologize to readers who've had to sit through my "8-Bit Lives" story before. And let me apologize in advance, because it's a story I suspect I'll be telling again and again. ■

## REFERENCES

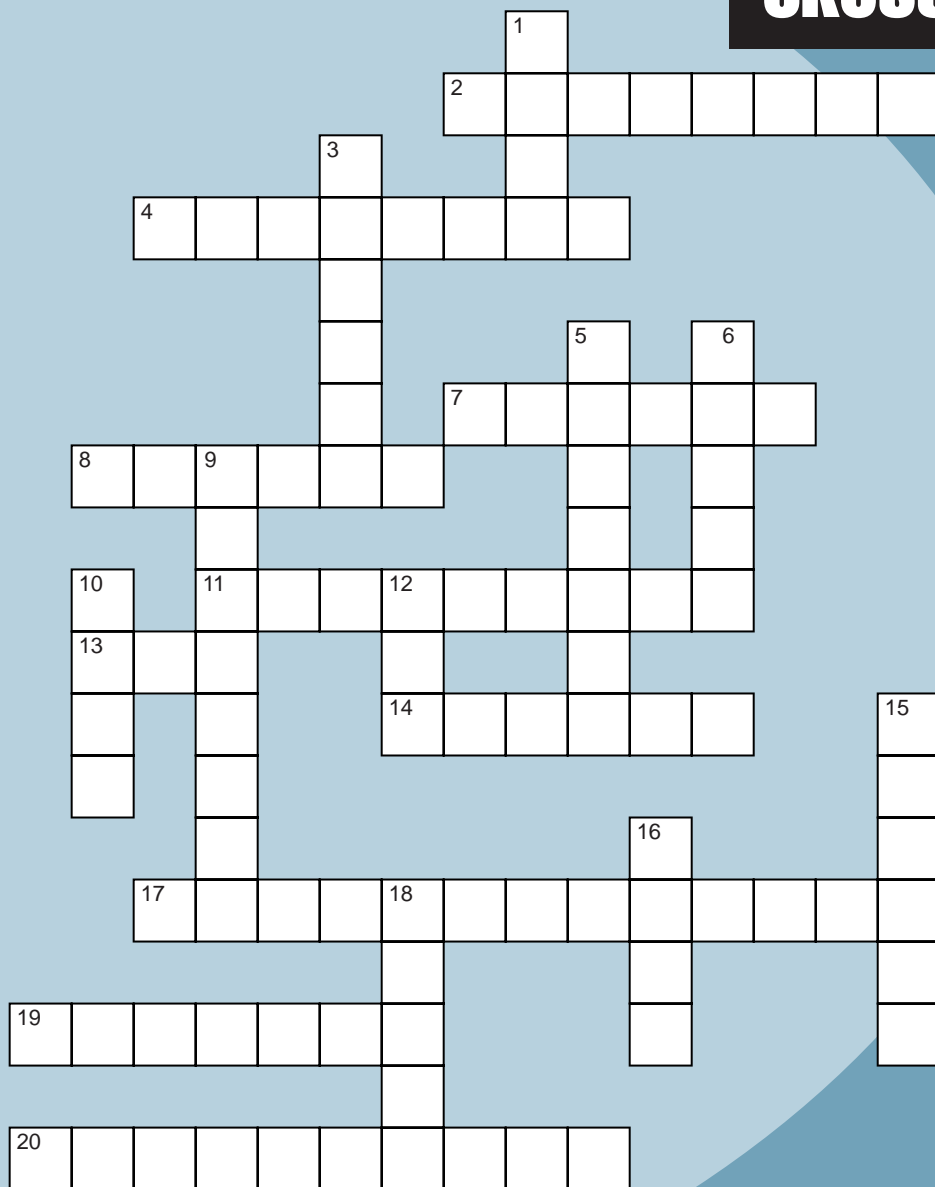
- [1] J. Shi, "AN3741: Implement an In-Application-Programmable Data-Storage on the MC9RS08KA8," Freescale Semiconductor, Inc., 2008.
- [2] V. Ko, "AN3266: Getting Started with RS08," Freescale Semiconductor, Inc., 2006.

## SOURCE

**MC9RS08KA, MC9RS08LA, and MC9RS08LE Microcontrollers**  
Freescale Semiconductor, Inc. | [www.freescale.com](http://www.freescale.com)



# CROSSWORD



## Across

2. `<B> ... </B>`
4. Transforms DC to AC
7. Seconds per cycle
8. Helical
11. Impedance Z
13. Live current!
14. Identifies wires
17.  $10^6$  Bq
19. Normal V
20. x,y

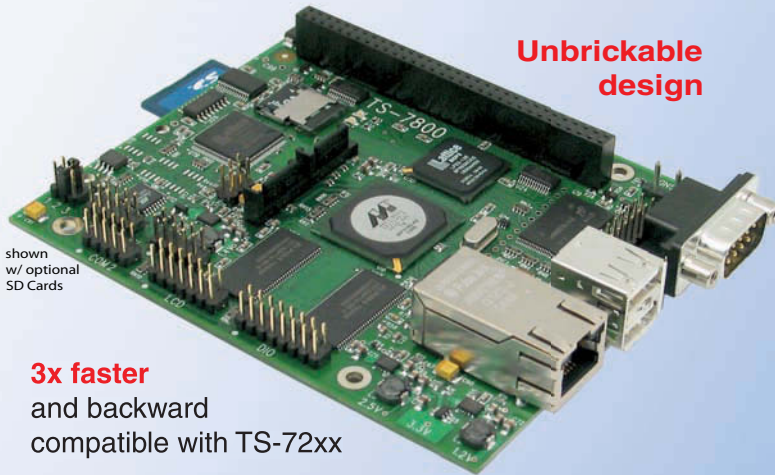
## Down

1. The "3" in the data sample: [9, 2, 3, 3, 3, 3, 8, 6, 6, 6, 1, 1]
3. CERN's city
5. The "O" in OLED
6. Mass  $\times$  acceleration
9. Init
10. Software added to a program to improve it
12. ASCII: End of transmission
15. The last color in the visible spectrum
16. Converter to drop V
18. Balanced/unbalanced

The answers are available at  
[www.circuitcellar.com/crossword](http://www.circuitcellar.com/crossword).

# Embedded Single Board Computers

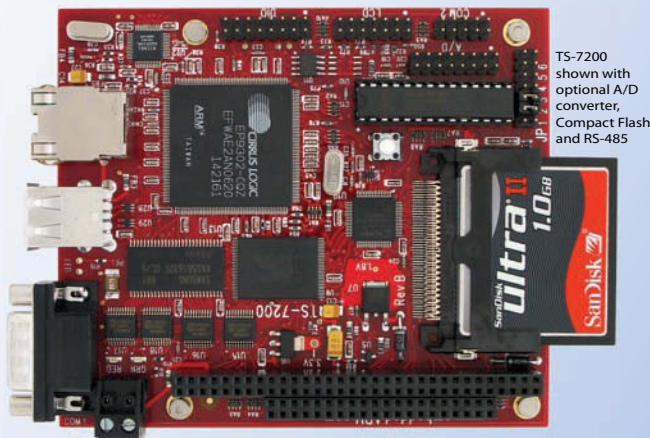
High-End Performance  
with Embedded Ruggedness



**TS-7800**  
**500 MHz ARM9**

- Low power - 4W@5V **\$229**  
qty 100
- 128MB DDR RAM
- 512MB high-speed (17MB/sec) onboard Flash **\$269**  
qty 1
- 12K LUT programmable FPGA
- Internal PCI Bus, PC/104 connector
- 2 host USB 2.0 480 Mbps
- Gigabit ethernet
- 10 serial ports
- 5 ADC (10-bit)
- Sleep mode uses 200 microamps
- Boots Linux in < 2 seconds
- Linux 2.6 and Debian by default
- 2 SD sockets
- 110 GPIO
- 2 SATA ports

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



**200 MHz ARM9**  
Power as low as 1/4 Watt

- 8 boards, over 2000 configurations **as low as \$99**  
qty 100
- Fanless, no heat sink
- SDRAM - up to 128MB **\$129**  
qty 1
- Flash - up to 128MB 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, Debian
- SD card option
- VGA video
- LCD ready

- options include:  
onboard temperature sensor, A/D Converter 8 channel 12 bit, Extended Temperature, Battery Backed Real Time Clock, USB Flash, USB WiFi

- 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

# Featured Products and PC/104 Peripherals

**NEW!**



**\$449**  
qty 1

## TS-TPC-7390

200MHz ARM9 Touch Panel Computer

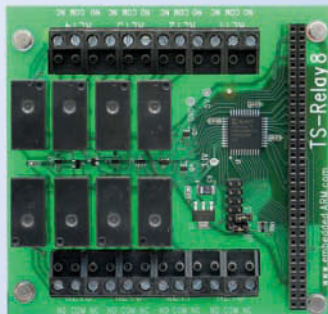
- Low Power, Industrial Quality Design
- Mountable aluminum frame
- 64MB SDRAM (128MB opt)
- 512MB Flash w/ Debian Linux
- Programmable FPGA- 5K LUT
- 7" Color TFT-LCD Touch-Screen
- 800x480 customizable video core
- Dedicated framebuffer- 8MB RAM
- Audio codec with speaker
- Boots Linux 2.6 in about 1 second
- Unbrickable, boots from SD or NAND
- Runs X Windows GUI applications
- Runs Eclipse IDE out-of-the-box

**NEW!**

## TS-RELAY8

Eight Software Controlled Relays

- 8 SPDT relays
- Up to 277 VAC @ 5A
- Up to 30 VDC @ 5A
- Software controlled
- 40mA draw per coil
- I/O jumpers



**\$89**  
qty 1

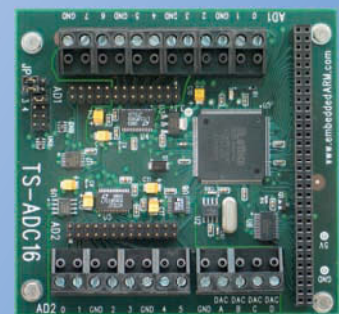
**NEW!**

## TS-ADC16

ADC, DAC and Digital I/O

- 16 16-bit ADCs
- Up to 100Ksps (10us)
- 1KB ADC RAM-FIFO
- 4 ADC voltage ranges
- Prog. pacing clock
- Externally triggered
- 4 inputs, 1 output
- 4 12-bit DACs
- 4 16 bit counters

**\$169**  
qty 1



see our website for x86 SBCs, peripherals and option details



We use our stuff.

Visit our TS-7800 powered website at  
[www.embeddedARM.com](http://www.embeddedARM.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" x 3" FORMAT.** Call for current rate and deadline information. E-mail [adcopy@circuitcellar.com](mailto:adcopy@circuitcellar.com) with your file and digital submission or send it to IDEA BOX, Circuit Cellar, 4 Park Street, Vernon, CT 06066. For more information call Shannon Barraclough at (860) 875-2199.

The Vendor Directory at [www.circuitcellar.com/vendor/](http://www.circuitcellar.com/vendor/) is your guide to a variety of engineering products and services.

## 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](http://www.dlpdesign.com)



NEW

## RS232 to TCP/IP

- TCP/com™ v2.0, RS232 to TCP/IP software. Plus TCP/IP to RS232.
- WinWedge™, RS232 or TCP/IP data direct into any Windows app. - Excel, Access, etc.

**TALtech**

Free 30 day evals at  
[www.taltech.com](http://www.taltech.com)

## 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.**

47808 Blvd. Henri-Bourassa  
Charlesbourg, Quebec, Canada G1H 3A7  
Tel: (418) 622-5298 • Fax: (418) 622-9996  
Email: [sales@picofab.net](mailto:sales@picofab.net)

## LPC3180 Dev Kit



The LPC3180 Development Kit provides a stable platform for building powerful user applications with the NXP Semiconductors LPC3180 within the ARM9 Linux environment.

The kit provides the user with an array of popular mobile technologies, making it an ideal development target for consumer communications software. Using existing Linux hardware drivers, developers can focus on building and improving their user interface applications without worrying about low-level hardware functionality.

The LPC3180 Development Kit may be purchased at Digkey.com, PN: 622-1018-ND or Mouser.com, P/N: 786-LPC3180-DEV-KIT, and includes the LPC3180 board, the LCD and Keypad Board, and everything else you need to get started.

FDI also provides low cost demo kits for NXP ARM7 LPC2xxx, LPC9xxx and ICP/ISP FLASH programmers.

**FDI**  
[www.teamfdi.com](http://www.teamfdi.com)  
VISA/MC/Amex

Future Designs, Inc.  
2702 Triana Blvd  
Huntsville, AL 35805  
(256) 883-1240  
Fax (256) 883-1241

## Control Your Motor

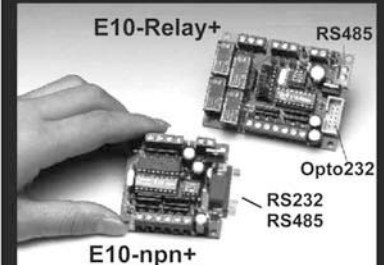


Use low-cost DC PM brush motors  
Proportional PWM Forward/Reverse  
Controls 2 motors independently  
Or mixed tank steering algorithm  
5-30 V, 30A/motor, 70A starting  
Adjustable current limiting  
Dynamic brake and Clutch output  
CDFR for 8 bit parallel port I/O  
RDFR for R/C servo command  
**(800) 882 6832** Visa/MC

**VANTEC.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](http://www.tri-plc.com/cci.htm)

Tel: 1-877-874-7527 - PLC specialist since 1993

**TRI**  
Triangle  
Research



**CIRCUIT CELLAR**  
FOR COMPUTER APPLICATIONS

Make sure you're signed up to receive Circuit Cellar's monthly electronic newsletter. *News Notes* will keep you up to date on Circuit Cellar happenings. Stay in the loop!

Register now. It's fast. It's free.  
[www.circuitcellar.com/newsletter/](http://www.circuitcellar.com/newsletter/)

### USB Data Acquisition

ADU208 -USB Relay I/O Interface

Complete SDK Online at: [www.ontrak.net](http://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](http://www.ontrak.net)

### PIC<sup>®</sup>MCU C Compiler

Version 4.100

**NEW**

- Optimized String Handling
- IDE Compilers include NEW Menu Manager

Supports NEW PIC16 Opcodes

Prices start at **\$150**

252.522.6500 x85 • [sales@ccsinfo.com](mailto:sales@ccsinfo.com)  
[www.ccsinfo.com/CCNEW](http://www.ccsinfo.com/CCNEW)



**Weather Instruments**  
for PCs

**AAG**  
electrónica

[www.aagelectronica.com](http://www.aagelectronica.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](mailto:sales@jkmicro.com)  
On the web at [www.jkmicro.com](http://www.jkmicro.com)

**JK microsystems**

### CROSSWARE<sup>®</sup>

Tools for Embedded Development

## ARM7 MODULAR TOOLSET

Add support for Atmel, NXP or STMicro' ARM7 variants to Base Package to suit your requirements and budget

- C/C++
- Code Wizards
- Debugging
- Simulation

Advanced software tools since 1984

[www.crossware.com](http://www.crossware.com)  
[info@crossware.com](mailto:info@crossware.com)

## 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](http://www.allelectronics.com)  
Free 96 page catalog  
1-800-826-5432

### 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!**  
[www.TraceSystemsInc.com](http://www.TraceSystemsInc.com)  
301-262-0300

**Trace SYSTEMS, Inc.**

## Solve complex signal acquisition problems...

- positioning & control
- environmental
- acceleration
- transients
- pressure
- vibration
- sonar
- GPS
- Linux Driver
- Guaranteed in stock
- Many newly added features
- 16-bit analog inputs and outputs
- Million sample FIFO eliminates interrupts
- Wide analog input and output ranges
- -40°C to +85°C Standard
- Order 24/7, fast and easy.



**www.stx104.com**  
Apex Embedded Systems  
help@stx104.com • 608-256-0767 x24



## Free Training for Rabbit Developers

Learn More at  
**Rabbit-U.com**

08078

## Wireless Analog !

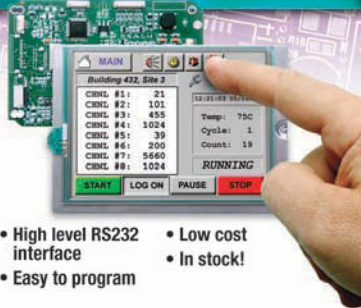


4-channel analog input wireless module  
433/868/915MHz. Low power, serial interface.  
Easy to configure. 24 bit ID to avoid cross-talk. 2  
digital output. Unique stamp-sized, v low power  
multipoint an/dig input 400/928MHz Tx/Rx  
module. Transmits to 50 feet. Ideal for toys,  
loggers, mobile eqt. 24-bit security. Networkable  
for relative position tracking.

EmbedRF Module \$59  
EmbedRF DevKit \$299

**Saelig** .com  
unique electronics 1-888-7SAELIG  
info@saelig.com  
www.saelig.com

## Add a color touch interface to your embedded product!



- High level RS232 interface
- Easy to program
- Low cost
- In stock!

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

## Adapt9S12XDP512 Modular Prototyping System

- \* Robotics and Mechatronics
- \* Electronic Fuel Injection
- \* Freescale 9S12XDP512
- \* RTOS-capable

Starting at \$125!



Evaluate \* Educate \* Embed

Program in  
Assembler, BASIC, C, and Forth

www.TechnologicalArts.com

**RENESAS**

www.renesas.com

## Going Wireless is Easy!

**Zigbee™**

### Wireless Mesh Network

NEW USB Zigbee™ Stick  
offers an easy way to Zigbee™ enable PC's

- ETRX2 USB Based on Ember Single Chip Zigbee™ 802.15.4 Solution
- No need for RF Design Experience
- 2.4 ISM Band



**LE MOS**  
INTERNATIONAL sales@lemosint.com

www.lemosint.com

## FlashCore-B (FB)™

Add embedded mass data storage, 16-bit ADCs to your product

\$79 Qty 1 \$34 OEM



- 2.1" x 2.4", C/C++ programmable.
- CompactFlash interface with FAT file system.
- 16-bit ADCs, DACs, RS232 and TTL I/Os.
- Ultra-low quiescent current for battery power.

50+ Low Cost Controllers with ADC, DAC, 18 UARTs, 300 I/Os, solenoid, relays, CompactFlash, LCD, DSP motion control. Custom board design. Save time and money.

**TERN**  
INC.

1724 Picasso Ave., Suite A, Davis, CA 95616 USA

Tel: 530-758-0180 • Fax: 530-758-0181

www.tern.com



sales@tern.com

# I<sup>2</sup>C/SMBus

- Bus Monitors
- Protocol Analyzers
- Host Adapters
- Multiplexers
- Battery Applications
- Software Tools

## MCC

Micro Computer Control

I<sup>2</sup>C is a trademark of Philips Corporation

[www.mcc-us.com](http://www.mcc-us.com)

# China PCB Supplier

## One-Stop Solution

Design - Fabrication - Assembly

- Instant Online Quotes - Day & Night
- On-time Delivery
- Prototype thru Production
- Reliable Quality
- UL Approved & ISO9001:2000 Certified Facility

**Even one piece, we care!  
Check out our low prices  
and save big \$\$.**

86 (571) 86795686  
sales@pcbcore.com

## PCBCORE.com

# PHYTEC

## phyCORE<sup>®</sup> SBCs Accelerate 32-bit Designs

phyCORE-LPC3250

COTS Single Board Computers:

- shorten time-to-market
- reduce development costs
- forgo substantial design issues and risks
- Windows<sup>®</sup> Embedded CE and Linux Board Support Packages (processor-dependent)
- \$170/unit benchmark price at 1K for ARM9-based SBC

**ARM:** i.MX31 (ARM11), i.MX27 (ARM9), LPC3250 (ARM9), LPC3180 (ARM9), LPC2294 (ARM7)

**XScale:** PXA320, PXA270, PXA255

**PowerPC:** MPC5554, MPC5200B, MPC565, MPC555

**ColdFire:** MCF5485    **Blackfin:** ADSP-BF537

- Rapid Development Kits start at \$399
- include SBC, Carrier Board, software, docu as well as kit-specific cables, adapters and LCD
- SBC module easily ports from Carrier Board to user target hardware
- Carrier Board serves as target reference design

[www.phytec.com](http://www.phytec.com) ■ 800.278.9913 ■ [www.phycore.com](http://www.phycore.com)

[www.can232.com](http://www.can232.com)

**Only \$108 €69**

**CAN232 Features:**

- Free sample programs
- 8-15VDC supply via CAN Timestamp in mS
- Small size 2.7" by 1.2"
- 100% Bandwidth up to 125Kbit
- Both 11 & 29 bit ID support
- 32 Message Receive FIFO
- Works up to 1Mbit CAN
- Simple ASCII protocol
- Supports RTR Frames
- Max 230Kbaud RS232
- Firmware upgradable
- No drivers needed
- OS Independent
- CE Approved

**CANUSB Features:**

- Free ActiveX component
- PC, MAC & Linux support
- Both 11 & 29 bit ID support
- Simple CAN logger included
- Free Threaded Windows DLL
- Firmware upgradable via USB
- Sample programs in C, C++, VB, Delphi, C#, PureBasic etc.
- No need for external power
- Works up to 1Mbit CAN
- Supports RTR Frames
- USB 2.0 Full Speed
- Free USB drivers
- CE Approved

**Only \$154 €129**

[www.canusb.com](http://www.canusb.com)

# ezLCD - The "Smart Display" makes integrating a GUI ez!

- \*Versatile Programming LCD module
- \*USB, SPI, RS232/TTL Interfaces
- \*Bright 350 Nit LED Display (320 x 240)
- \*Integrated Touch Screen
- \*Lua Scripting Language capable - For stand alone embedded apps
- \*Memory 3.8 MB + SD to 2G
- \*ezLCD's are also available in 2.7", 5.6", 6.4", 8.0", 10.4"

3.5" ezLCD p/n: ezLCD-103

Call for Custom Display Configurations

[www.earthlcd.com](http://www.earthlcd.com)

# LV- MaxSonar

## Ultrasonic Ranging is EZ

### LV-MaxSonar Products

- High quality • Low cost
- Low power, 3V-5.5V, (< 4mA avg.)
- Easy interfacing • Serial, pulse width, & analog voltage outputs
- Reliable and stable range measurement • No dead zone

### LV-MaxSonar-EZ

- Choice of beam patterns
- Tiny size (<1 cubic inch)
- Light weight (<5 grams)

### LV-MaxSonar-WR1 (IP67)

- Industrial packaging
- Weather resistant
- Standard 3/4" fitting
- Quality narrow beam

[www.maxbotix.com](http://www.maxbotix.com)

# IMAGEcraft

## development tools for:

### hardware kits:

**NEW: the eBox- AVR!**  
Perfect for students and hobbyists

**Elmicro kits - HC08 & HCS12 / 12 ref. design & starter kits**

**Ethernut - Complete AVR TCP/IP solution**

Version 7 Tools  
Full-featured 45-day demos on our website!

starting at **\$249!**  
( \$99 for students )

Since 1994

[www.imagecraft.com](http://www.imagecraft.com)  
info@imagecraft.com  
(650) 493-9326 • FAX (650) 493-9329

VISA / MC / AMEX  
Discover / Check

- Mechatronics and Robotics
- Advanced 16-bit MCU
- RTOS-capable

**Get started for as little as \$29!**

|                   |                      |
|-------------------|----------------------|
| SCI               | 32K Flash            |
| SPI               | 2K RAM               |
| Vreg (3.3V to 5V) | 8 ch. 10-bit A-to-D  |
| Key Wakeup Ports  | 16-bit Timer         |
| Background Debug  | up to 6 channels PWM |
| CAN 2.0 A/B       |                      |

HCS12 CPU

**Program in Assembler, BASIC, C, and Forth**

[www.NanoCore12.com](http://www.NanoCore12.com)

## HIGH PERFORMANCE SOCKETS & ADAPTERS



- Package Converters SOIC/DIP, BGA/QFP, BGA/BGA and more
- 40 GHz Bandwidth BGA/QFN Sockets
- Heat Sinks available up to 100W
- Adapters for probing/test/prototype
- High Volume Production Adapters-BGA, QFP and more
- SMT Package Emulation/Interconnect

**Quick-Turn Complex Custom Adapters**

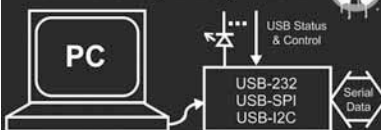


**Ironwood Electronics**

**1-800-404-0204**

[www.ironwoodelectronics.com](http://www.ironwoodelectronics.com)

## USB-to-serial chips Low cost, Easy R&D



- Add USB to your products in a day
- No drivers needed for Windows, Mac, Linux
- No microcontroller programming required
- Check out the whole range at HexWax e.g. make your product a Flash Drive!

[www.hexwax.com](http://www.hexwax.com) - Mouser - Farnell - Digkey

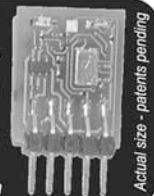
## Amazing PIC programmer

Most devices supported

ICSP, SQTP, & copy limits

**\$32** at Digkey & Mouser

[www.flexipanel.com](http://www.flexipanel.com)



Actual size - patents pending

## 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](http://www.spectraplus.com)

**PHS**

Pioneer Hill Software  
360 697-3472 voice  
pioneer@telebyte.com

## Powerline Communication Modem



230VAC/50Hz, 110V  
AC/60Hz powerline  
or 12V/18V/24V DC  
powerline as signal  
wire.

Interfaces: RS485, RS232, UART (Serial TTL)

Built-in repeater function to extend range

- Fully transparent mode, plug and play coming out of the box without the need to do any programming.
- Built-in error correction codes.
- Physical and logic address, support addressing and broadcasting.
- AT commands used for advanced configuration.
- Up to 19.2kbps



Powerline to Zigbee

 LinkSprite

[www.linksprite.com](http://www.linksprite.com)

## PRINTED CIRCUIT BOARDS

QUALITY PRODUCT

FAST DELIVERY

COMPETITIVE PRICING

- 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

**PROTOTYPE**  
through  
PRODUCTION

### 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](http://www.pulsar-inc.com) • [sales@pulsar-inc.com](mailto:sales@pulsar-inc.com)



Order online at:  
[www.melabs.com](http://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

**\$89.95**  
(as shown)

RoHS  
Compliant

Programs  
PIC MCUs  
including  
low-voltage  
(3.3V) devices

Includes  
Software for  
Windows 98,  
Me, 2K, & XP



With Accessories for \$119.95:  
Includes Programmer, Software, USB Cable,  
and Programming Adapter for 8 to 40-pin DIP

### Serial LCDs

2-line x 16 \$39.95  
4-line x 20 \$49.95

Quantity Discounts  
Available!



### 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**

32-bit signed variables and math operations  
Supports Microchip PIC10, PIC12, PIC14,  
PIC16, PIC17, and PIC18 microcontrollers  
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](http://www.melabs.com)



# INDEX OF ADVERTISERS

The Index of Advertisers with links to their web sites is located at [www.circuitcellar.com](http://www.circuitcellar.com) under the current issue.

| Page                              | Page                        | Page                             | Page                                      |
|-----------------------------------|-----------------------------|----------------------------------|---|
| 75 AAG Electronica, LLC           | 20 ExpressPCB               | 13, 76 Lemos International       | 78 Pulsar, Inc.                           |
| 32 AP Circuits                    | 10 ezPCB                    | 78 LinkSprite Technologies, Inc. | 41, 48 Rabbit, A Digi International Brand |
| 75 All Electronics Corp.          | 74 FDI-Future Designs, Inc. | 24 Linx Technologies             | 76 Rabbit, A Digi International Brand     |
| 19 Altium Ltd.                    | 78 FlexiPanel Ltd.          | 5 Luminary Micro                 | 76 Reach Technology, Inc.                 |
| 76 Apex Embedded Systems          | 11 Front Panel Express LLC  | 77 MCC (Micro Computer Control)  | 76 Renesas Technology                     |
| 25, 27 ARM                        | 28 General Circuits         | 45 MachinePIER                   | 49 RoboBusiness Expo                      |
| 29 ARM                            | 13 Grid Connect, Inc.       | 77 Maxbotix                      | 76 Saelig Co.                             |
| 7 Atmel                           | 39 HI-TECH Software LLC     | 78 microEngineering Labs, Inc.   | 12 Sealevel Systems                       |
| 33 CWAV                           | 69 HobbyLab LLC             | 2 Mouser Electronics             | 9 SEGGER Microcontroller Systems LLC      |
| 57 CadSoft Computer, Inc.         | 47, 50 ICbank Inc.          | C2 NetBurner                     | 55 Sensors Expo & Conf.                   |
| 34 Calao Systems                  | 77 IMAGEcraft               | 3 Noritake Co., Inc.             | 74 TAL Technologies                       |
| 63 Comfile Technology, Inc.       | 1 Imagineering, Inc.        | 37 Nurve Networks LLC            | C3 Tech Tools                             |
| 75 Crossware Products, Inc.       | 78 Ironwood Electronics     | 75 Ontrak Control Systems        | 72, 73 Technologic Systems                |
| 75 Custom Computer Services, Inc. | 32, 34 JKmicrosystems, Inc. | 77 PCBCore                       | 76, 77 Technological Arts                 |
| 74 DLP Design                     | 75 JKmicrosystems, Inc.     | 17 PCB-Pool                      | 76 Tern, Inc.                             |
| 69 DesignNotes                    | 30 Jameco                   | C4 Parallax, Inc.                | 75 Trace Systems, Inc.                    |
| 11 EMAC, Inc.                     | 37 Jeffrey Kerr, LLC        | 77 Phytec America LLC            | 74 Triangle Research Int'l, Inc.          |
| 42 ESC-West                       | 58 Keil Software            | 74 Picofab Inc.                  | 68 Trinity College Robot Contest          |
| 77 Earth Computer Technologies    | 8 LPFK Laser & Electronics  | 78 Pioneer Hill Software         | 74 Vantec                                 |
| 48 Electronicstalk                | 37 Lakeview Research        | 65 Pololu Corp.                  |   |
| 24 Elprotronic                    | 77 Lawicel AB               | 45 PROPOX Sp. z o.o.             |   |

## PREVIEW of April Issue 225

Theme: **Embedded Programming**

**Construct a USB GPIO Pod (Part 1):** No Parallel Port, No Problem

**Robot Navigation and Control (Part 2):** Software Development

**Digital Decoding:** A Design for Decoding Periodic Signal Transmissions

**THE DARKER SIDE Time Domain Reflectometry:** Detect, Measure, and Locate Impedance

**ABOVE THE GROUND PLANE Solar Data Capture:** PCB Layout Woes, Inductor Design, and More

**FROM THE BENCH Programmable Robotics (Part 2):** Application Development

**SILICON UPDATE ZStar Trek:** A Healthy Mix of MCUs, Sensors, and Wireless Technology

### ATTENTION ADVERTISERS

#### May Issue 226 Deadlines

Space Close: Mar. 13  
Material Close: Mar. 20

**Theme**  
**Measurement & Sensors**

**Bonus Distribution**  
Sensors Expo

Call Shannon Barraclough  
now to reserve your space!  
**860.875.2199**

e-mail: [shannon@circuitcellar.com](mailto:shannon@circuitcellar.com)

# PRIORITY INTERRUPT



by Steve Ciarcia, Founder and Editorial Director

## Cloud Computing

**Y**ou know, a lot of what people are describing today as the future of the Internet sounds like déjà vu to me. Way back when, I remember having dumb terminals all over the place that time-shared the services of a remote-located mainframe computer. I loaded my data into programs that were batch processed on the central computer and then sent back to the terminals. While highly effective when it came to getting more computing power than a slide ruler, it was terribly ineffective when it came to time management, data privacy, and asset control. It's no wonder why we all zealously jumped on the distributed processing bandwagon of personal computing. At least then we were in complete control, limited only by how much processing power we could jam in the box on our own desk. I call that progress.

Unfortunately, technology has now progressed to the point where we seem to be heading in reverse again. By whatever lack of historical perspective, people are thinking that big service companies, shared software applications, and remote computing are better than local services and local control. This isn't a rant. Personally, I see benefits and detriments on both sides of the argument.

Specifically, I'm talking about cloud computing and its ramifications. While some use this term to describe virtual client servers on the Internet, I prefer the more accepted description that cloud computing is all the real-time software and subscription-based services that you use via the Internet. This includes things like Google Docs, virtual data storage, video viewing/posting services, Google Maps, etc., etc. It's a long list and the "cloud" gets bigger every day.

I attribute the move toward cloud computing as an addictive escalation in all the connectivity that everyone seems to be seeking. It's not enough to just have a cell phone number where we can be reached any more. Now we have to have constant live downloads to steroid-stuffed communication devices designed to enhance every aspect of the things we see, touch, or do. Look at a subway map and all the schedule information suddenly streams from the cloud into our Apple iPhone. At the same time our GPS location goes out to the cloud along with "tweets" (i.e., posts on Twitter), text messages, and blog updates that make a private life everything but. And, unlike we skeptics, devotees of the new faith have no fear of completely trusting all the software services in the cloud. Of course, it doesn't hurt that most of these applications are free but, at this point in its evolution, I wonder how many of these free services are really virtual heroin. ;-)

The other side of the equation is that putting all your computing capability in assorted boxes is complex and expensive. When we only had a single desktop, it was less complex because we needed only a single software license for the operating system and programs we used. As we progressed to having a couple of desktops and two or more laptops per house (last count I had six total), the issue became the inordinate cost involved with all the single-platform licenses, software updates, and communication services needed to support the "farm." Enriching Microsoft at every turn isn't a crowd pleaser any more than my having to buy six copies of the same virus protection software every year.

The attractiveness of the uncorrupted cloud concept is that people can supposedly use lower-function computers (i.e., cheaper) with perhaps an open-sourced OS and no single license-indigenous applications. Instead, each box would simply connect to web sites in the cloud (think virtual mainframes) and execute subscription or pay-by-use application software, as needed. Instead of purchasing schematic design programs or Microsoft PowerPoint for each computer, companies simply sign on to application resources and do everything online for pennies. This is nirvana for a medium-sized outfit facing the purchase of a couple hundred new desktops for its staff; but it is certainly a bummer if you live in Redmond, WA, and work for a company trying to think up new ways of peddling more copies of Microsoft Vista and Microsoft Office (grin).

Good or bad, acceptance and extensive utilization of cloud computing in our future is as inevitable as multicore processors, because the more computer applications we exercise, and the more social connectivity we enjoy, the more everyone wants. I don't see myself starting a MySpace page and Twittering readers between bites at lunch, but I do see the merits in being able to try many more software applications simply because using a virtual mainframe eliminates all the crash-and-burn terror of physically adding new software to an otherwise stable PC.

The caution is that cloud computing is still a cute and cuddly concept in its infancy, but it can be easily monopolized or corrupted if we don't look at history when guiding the future. Fanatical acceptance and hasty growth of cloud computing is being driven by an abundance of free and low-cost applications that, in my opinion, are all about market share. Beware that we may be trading one land-based monopoly for new monopolies in the cloud based on computing power and services (think Amazon and Google). I'll be back.

steve.ciarcia@circuitcellar.com

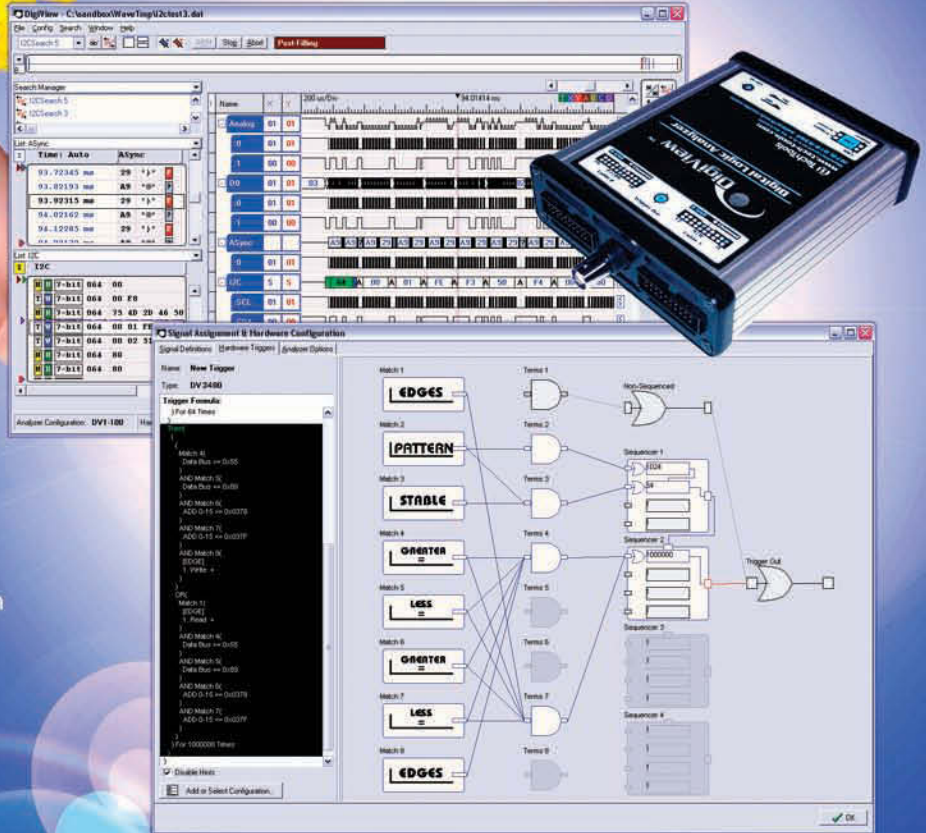
A handwritten signature in black ink, appearing to read "Steve".

# PC Based Logic Analyzers

**NEW Model DV3400**

from \$499.00

- Faster Sampling (200/400 Msps)
- More Channels (36)
- More Memory (4x)
- Advanced Match circuits (8)
- ±6V Adjustable Threshold (x2)
- Cascadable Sequencers (4)
- Enhanced Compression



## Professional Hardware Capture + Software Analysis

- Automatic Real-time Hardware Compression eliminates the need to reduce resolution. Our newest version makes "dead time" insignificant.
- Edge and Pattern Triggers on all models. Advanced Model also includes Range (>, <, =, !=, >=, <=) and Stable Matches with Duration, and 4 flexible cascadable sequencers with pass-counters.
- Pattern Searches with Match & Duration.
- Specialized Sequential Searches for Serial and State Mode Signals.
- Display I<sup>2</sup>C, Synchronous (SPI), Asynchronous (RS-232), State, Boolean, Bus and Analog Data.
- Single or Dual Waveform Views.
- Resolution Zoom in Wave Views.
- Time based Link Groups for all views.
- Specialized Exports from Data Tables and List Views.
- Multi-Signal Data Tables.
- Drag & Snap Markers.
- "Click to Center" function.
- "Snap Previous/Next" function.
- Print or Save Images with comments.
- USB 2.0 (480 Mbps).
- USB 1.1 compatible (12 Mbps).

Very flexible, "easy to configure" Advanced triggering.

## PICmicro<sup>®</sup> MCU Programmer

Multi-Function, In-Circuit & Gang Operation

only \$199.00



See our Memory Emulators  
from \$179.00

**TechTools**  
www.tech-tools.com  
(972) 272-9392 • sales@tech-tools.com

# Robots

Your search is over.



Order **Parallax Robots and Accessories** at [www.parallax.com](http://www.parallax.com) or call our Sales Department toll-free at 888-512-1024 (Mon-Fri, 7 a.m. - 5 p.m., PT).

Parallax, and the Parallax logo are trademarks of Parallax Inc.

**PARALLAX**   
[www.parallax.com](http://www.parallax.com)