#201 April 2007

www.circuitcellar.com

CIRCUIT CELLAR®

THE MAGAZINE FOR COMPUTER APPLICATIONS

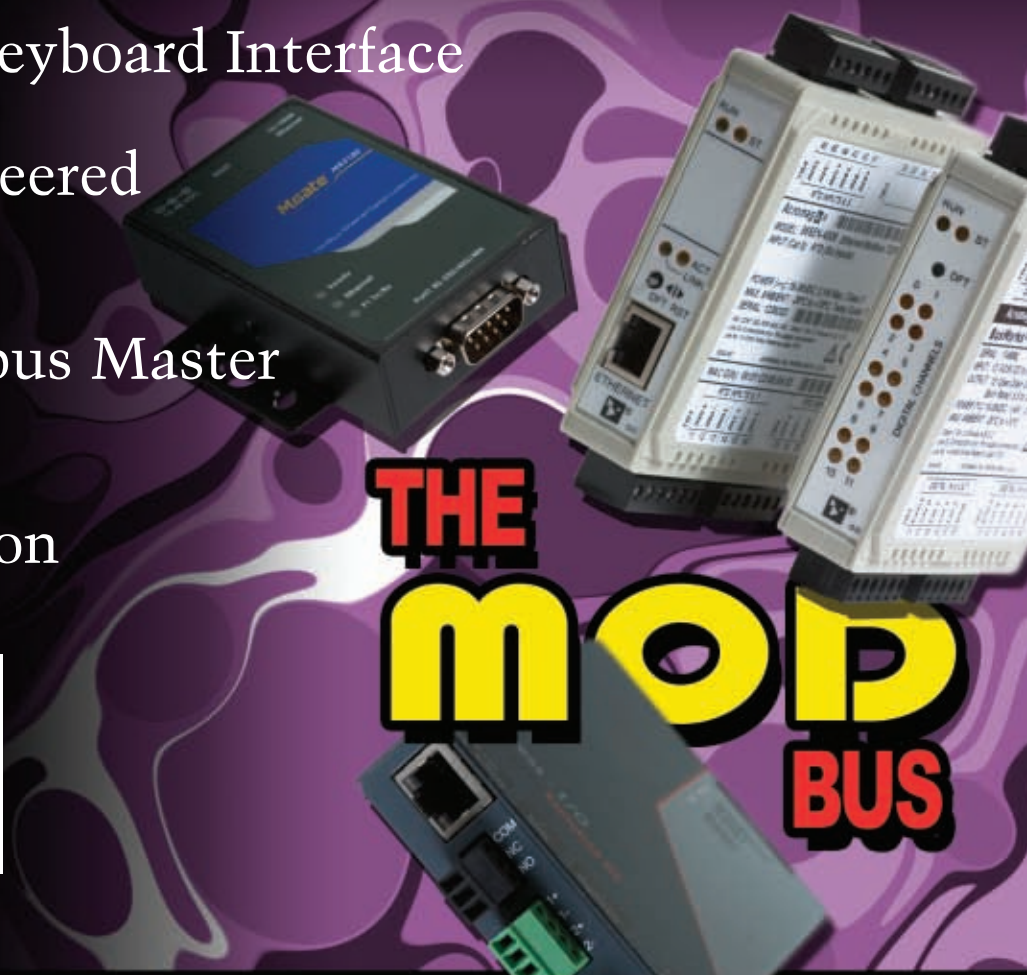
EMBEDDED PROGRAMMING

MCU-Based Keyboard Interface

Reverse-Engineered ECP Bus

Create a Modbus Master

Embedded RF Communication

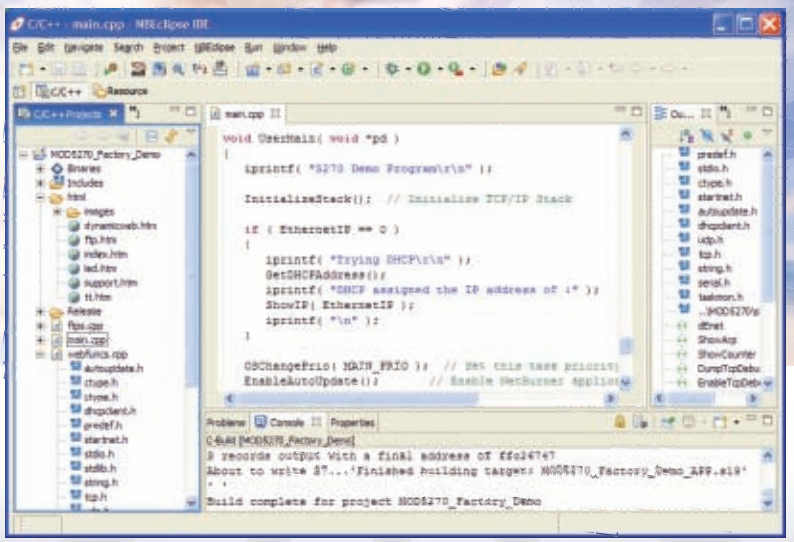


04>
7 25274 75349 9
$4.95 U.S. ($5.95 Canada)

The NetBurner Eclipse Ethernet Development Kit

Only **$99**





Featuring NetBurner's Eclipse IDE!

One Click Compile and Load | Intelligent Code Completion | Integrated Debugger

The Complete Hardware and Software Solution

Includes NetBurner MOD5270 Ethernet Core Module

Performance and Memory | 32-Bit CPU | Freescale ColdFire MCF5270 147Mhz | 512K Flash Memory | 2MB SDRAM

Device Connectivity | 10/100 Ethernet | 3 UARTs | I²C | SPI | 47 Digital I/O | SD/MMC Flash Card Support

Communication Software

| TCP/IP Stack | HTTP Web Server | FTP | E-Mail | PPP | Flash File System

Development Software

| NB Eclipse IDE | Graphical Debugger | Deployment Tools | Examples

System Software

| µC/OS RTOS | ANSI C/C++ Compiler and Linker

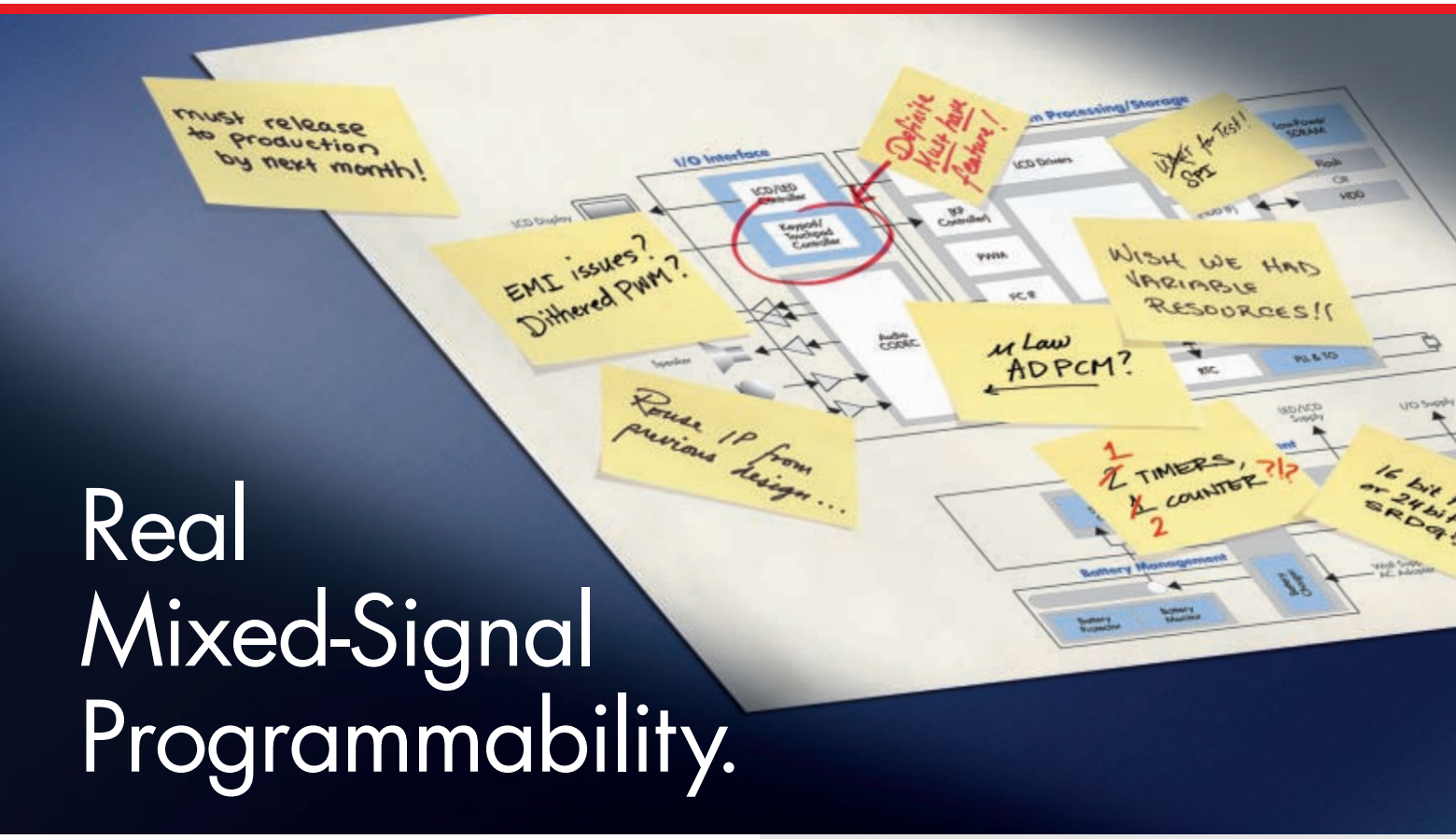
NetBurner.com

freescale semiconductor

Product No. | NNDK-MOD5270LC-KIT

Information and Sales | sales@netburner.com

Telephone | 1-800-695-6828



Real Mixed-Signal Programmability.

Get PSoC®. Because change happens.

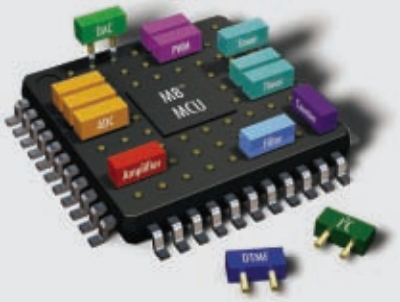
PSoC flexibility enables changes anytime: at concept, through production, in the field. Specifications change constantly. Yet pressures to differentiate, minimize costs, and speed time-to-market remain the same. To stay ahead of the curve, you need flexibility, programmability, and scalability. PSoC's unique programmable architecture delivers this and more. Futureproof your design; make PSoC your agent of change.

PSoC delivers:

- The configurability of an FPGA, the mixed-signal integration of an ASIC, and the familiarity of an MCU.
- Reusable IP, compatible device families and variable resource options ensure you can optimize design efforts and accommodate changes.
- The industry's first visual embedded design tool, PSoC Express™, speeds design time, enabling you to generate a complete design without writing a single line of code.

GET STARTED WITH PSoC NOW.

- *Download our "Change Happens" White Paper and get 50% off a PSoC development kit:* **www.cypress.com/changepaper**
- *Download free PSoC Express™ visual embedded software:* **www.cypress.com/changesoft**
- *Request free PSoC device samples:* **www.cypress.com/changechip**
- *Free online training:* **www.cypress.com/changetrain**
- *Purchase PSoC development tools:* **www.cypress.com/changetools**



PSoC includes programmable analog and digital blocks, a fast MCU, flash and SRAM memory, all in a compact package (as small as 3mm x 3mm).

CYPRESS PERFORM

© 2007 Cypress Semiconductor Corporation. All rights reserved. PSoC is a registered trademark and PSoC Express is a trademark of Cypress Semiconductor Corporation. All other trademarks are properties of their respective owners.

Link Instruments

PC-Based Test Equipment

Digital Oscilloscopes



- 2 Channel Digital Oscilloscope
- **500 MSa/s** max single shot rate
- 1Mpt sample memory
  - 250 MSa/S (Dual channel) 512 Kpts
  - 500 MSa/S (Single channel) 1 Mpts
- Advanced Triggering
- Only 9 oz and 7" x 3.5" x 1.5"
- Portable and Battery powered
- USB 2.0
- Advanced Math
- FFT Spectrum Analyzer
- $950 (DSO, Probes, Software & power supply)



Windows Screenshot

Logic Analyzers



Windows Screenshot



LA5240 (200MHz, 40CH)	**$1700**
LA5280 (200MHz, 80CH)	**$2350**
LA5540 (500MHz, 40CH)	**$2500**
LA5580 (500MHz, 80CH)	**$3500**
LA55160 (500MHz, 160CH)	**$7500**

- 40 to 160 channels
- up to 500 MSa/s
- Variable Threshold
- 8 External Clocks
- 16 Level Triggering
- up to 512K samples/ch
- USB 2.0 and Parallel Interface
- Pattern Generator option

Link Instruments (973) 808-8990

17A Daniel Road East · Fairfield, NJ 07004 · Fax (973) 808-8786

www.Linkins4.com

For The Premium Application

that took years to develop and must be instantly read under all lighting, temperature, and viewing angle conditions:

Introducing the New

3900 VFD Series



256 x 128 format now available in three sizes

THE MOST POWERFUL VACUUM FLUORESCENT GRAPHICS MODULE EVER OFFERED

- Multiple Built-in Windows
- Many International Fonts
- Font Magnification
- Large Memory
- High Density Pixels
- Multiple Interfaces
- Highest Brightness
- Extended Temperature Range
- Widest Viewing Angles
- Choose From 11 Models

Noritake

100 years of Quality Products & Services

www.noritake-elec.com/52

Over 300 standard VFD modules available

inventor of the vacuum fluorescent display

TASK MANAGER

The Cruelest Month?

Even if you spent your college years working with circuit boards and writing code, you're probably familiar with the famous first sentence of the poet T.S. Eliot's modernist masterpiece, *The Waste Land* (1922):

"April is the cruelest month—"

You may have heard a few exhausted bookkeepers and accountants mumble that line a few times too. Tax season, right?

Well, we here at *Circuit Cellar* have a different take on April. It's a great month. Not only are we now past the days of sleet, snow, and dark commutes to and from the office, we have a lot of exciting things on our plate—which is your plate too, of course.

This month we're off to the Embedded Systems Conference in San Jose, where we plan to spend the entire week talking everything embedded with our readers and advertisers. Look for our booth. We hope to see you there!

The Microchip 16-Bit Embedded Control Design Contest also kicks off this month. As you know, we've brought you a lot of amazing contests over the years. This one is sure to create a lot of buzz in the design community. With a handful of amazing eligible parts to choose from, this contest is going to get competitive quick. Ready to hit the workbench?

On the editorial front, we're excited about the articles in this issue. On page 12, we begin with an article about Steven Savage's award-winning ATir keyboard interface design. This project won him First Prize in the Atmel AVR Design Contest 2006. If you're preparing a project for the new Microchip contest, be sure to check out this article. Steven shows what it takes to put together a winning project.

Another project-inspiring design is Tom Kibalo's Java-based 1-Wire distributed watering system (p. 20). Tom used his hardware and software skills to build a handy system to water plants in six EarthBoxes. Even if gardening isn't your thing, you can use the principles he covers to design and program a multiple-module system for other purposes.

At the end of his article about the watering system, Tom writes that he plans to use ZigBee technology to extend the system's range beyond its 1-Wire limitations. Well, you don't need to wait until Tom upgrades his system and writes another article. Why not take Fred Eady's advice and implement the technology in a system of your own? In "Uncomplicated RF Communication" (p. 45), Fred shows you how to get your system on the air.

On page 28, Gordon Dick describes how he designed and built a three-axis CNC mill machine. The easy-to-move machine is a great project for anyone who loves design and woodworking. It features an old laptop, an intelligent motion controller, and an old X-ray machine.

Have you tried Visual Basic 2005 yet? As you know, Aubrey has been putting it to use for some time now. This month, he finishes up his series about a generic Modbus simulator (p. 60). To put you on the right track, he explains how he created the generic interface to gain access to any Modbus slave via an Excel workbook. As Aubrey explains, the application simply pulls the information from Excel, presents it on the UI, formats it, and sends and receives Modbus messages.

With projects like these to keep you occupied, it's going to be a productive month.

cj@circuitcellar.com



CIRCUIT CELLAR®

THE MAGAZINE FOR COMPUTER APPLICATIONS

FOUNDER/EDITORIAL DIRECTOR
Steve Ciarcia

MANAGING EDITOR
C.J. Abate

WEST COAST EDITOR
Tom Cantrell

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

NEW PRODUCTS EDITOR
John Gorsky

PROJECT EDITORS
Steve Bedford
Ken Davidson
David Tweed

ASSOCIATE EDITOR
Jesse Smolin

CHIEF FINANCIAL OFFICER
Jeannette Ciarcia

MEDIA CONSULTANT
Dan Rodrigues

CUSTOMER SERVICE
Debbie Lavoie

CONTROLLER
Jeff Yanco

ART DIRECTOR
KC Prescott

GRAPHIC DESIGNER
Mary (Turek) Sobuta

STAFF ENGINEER
John Gorsky

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

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

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

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

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

PRINTED IN THE UNITED STATES

CONTACTS

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

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

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

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

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

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

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

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

Entire contents copyright © 2007 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.

NKK SWITCHES

PRODUCTION EFFICIENCY

ILLUMINATED SOLUTIONS IMPROVE PROJECT TIMELINES

Your production efficiency will shine with our brightly illuminated solutions. NKK's SmartSwitch™ allows you to improve your processes by providing simple solutions to complex procedures. Our series of LCD programmable switches and displays feature high-contrast and wide viewing angles to simplify and accelerate your project timelines.



IS Dev Kit-2

SMARTSWITCH™

ACTIVATIONS MADE SIMPLE

Get Your FREE 3D CAD Model at:

http://cc.nkksmartswitch.com

Power your designs with NKK. Visit us online to:

- **ORDER** Dev Kits to drive your creativity
- **DOWNLOAD** software to expedite your design process
- **PROTOTYPE** and test new design ideas cost effectively

April 2007: Embedded Programming

FEATURES

12 **ATir Keyboard Interface**
Steven Savage
First Prize Atmel AVR Design Contest 2006

20 **Java-Based EarthBox Watering System**
Tom Kibalo

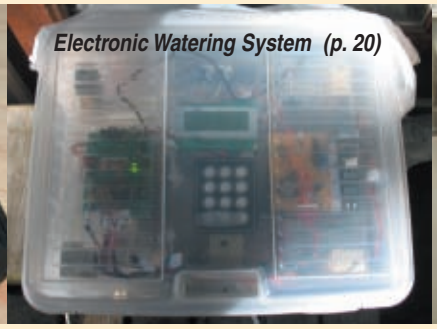
28 **Build a Three-Axis CNC Mill Machine**
Gordon Dick

38 **Reverse-Engineered ECP Bus**
Miguel Sanchez

45 **Uncomplicated RF Communication**
Fred Eady

60 **Generic Modbus Simulator (Part 2)**
Create a Modbus Master Using Visual Basic 2005
Aubrey Kagan


ATtiny45-Based Keyboard Interface (p. 12)


Electronic Watering System (p. 20)

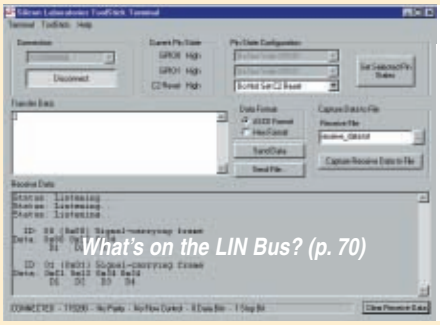

Computer-Controlled Mill Machine (p. 28)

COLUMNS

56 **ABOVE THE GROUND PLANE**
Battery Capacity Charge
Ed Nisley

70 **FROM THE BENCH**
Local Interconnect Network
Jeff Bachiochi

78 **SILICON UPDATE**
USB: (Wire)Less is More
Tom Cantrell


What's on the LIN Bus? (p. 70)


USB Unplugged? (p. 78)

DEPARTMENTS

4 **TASK MANAGER**
The Cruelest Month?
C.J. Abate

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

93 **CROSSWORD**

94 **INDEX OF ADVERTISERS**
May Preview

96 **PRIORITY INTERRUPT**
Only Real Events Lead to Real Trends
Steve Ciarcia



AVR picoPower Microcontrollers

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

What can AVR picoPower do for your design?

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

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

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

Everywhere You Are®

NEW PRODUCT NEWS

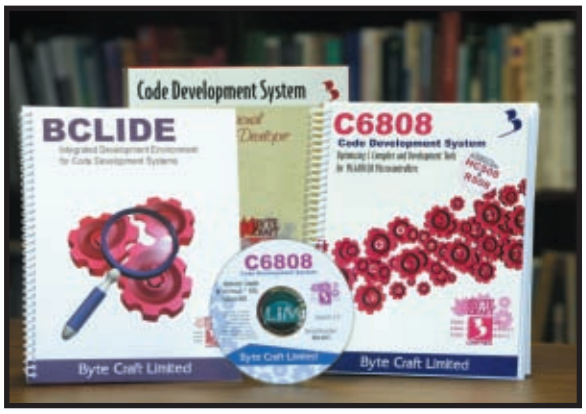
Edited by John Gorsky

RS08 C COMPILER

The **C6808 Code Development System** that supports Freescale Semiconductor's advanced 8-bit RS08 microcontroller architecture is now shipping. The C6808 is one of the first C compilers for RS08. It allows easy migration of programs written for Freescale Semiconductor's HC05 and HC08 to the high-volume 8-bit microcontroller platform. The C6808 supports all HC08, HCS08, and now RS08 parts in its optimizing C compiler and tools.

The C6808 Code Development System includes an optimizing C cross-compiler with an inline macro-assembler, support libraries, full program optimization with the BClink optimizing linker, and the BCLIDE. The compiler produces industry-standard executables and code files, including the Byte Craft Limited .COD file format and ELF/DWARF. The .COD file and ELF/DWARF contain everything an emulator needs for C-source-level debugging.

The C6808 supports ISO TR 18037 named address spaces and fixed-point math. It implements the full C language within the limits of the hardware: Any potential application can be implemented in C with greater ease and optimization than in assembly. Thread support in C complements the RS08 interrupt architecture and threads can model hardware interrupt arbitration in software. They typically allow faster, smaller code and less RAM consumption than conventional asynchronous programming.



The C6808 costs **$1,750**.

Byte Craft Limited
www.bytecraft.com

MODULAR KICKSTART KIT TARGETS R8C, M16C, AND M32C

The new **IAR0018** KickStart kit contains all the necessary tools for initial evaluation and prototyping on the R8C/M16C/M32C microcontroller range, giving developers the opportunity to test 8-, 16-, and 32-bit microcontrollers within the same core family. It also provides the flexibility to change the microcontroller during the course of the evaluation. Using the same fully featured tools, which will be used when real product development starts, makes it possible to avoid the delays and errors that can be caused by changing tools and the development environment after the evaluation proof-of-concept milestone is reached.

The kit includes code-limited versions of the Embedded Workbench for the M16C, the M32C, and the R8C. An application baseboard fitted with a microcontroller from one of seven available device families such as the R8C, the M16C, or the M32C is also included. The board is fitted with three serial interfaces: two MAX232s and one TTL along with a USB-to-serial interface, CAN with a transceiver, user LEDs, a 16 × 2 character LCD module with an LED backlight, an LDR and potentiometer connected to the A/D-channel, an encoder switch with a push button, a MMC/SD card interface, and other features.

The kit also includes a Renesas Technology E8-USB debugger, with all the necessary cables and documentation. It works for all the relevant microcontrollers and can be driven from within an IAR Embedded Workbench. The IAR0018 costs **$495**.



IAR Systems
www.iar.com

VGA CONTROLLERS FOR EMBEDDED APPLICATIONS

The **ezVGA Graphics Controller** and **ezVGA Serial Module** are two new display controllers from Multilabs. The ezVGA Graphics Controller is an embedded graphics controller that can be interfaced to any design via an 8-bit data bus and four addressable control lines. Through the ezVGA Graphics Controller, your embedded controller or processor will easily generate color VGA graphics. It provides all the circuitry necessary for a host interface, video memory control, and generating color VGA graphics with high screen resolution.



The ezVGA Serial Module takes it one step further. With the ezVGA Graphics Controller and an on-board microcontroller, it scales high-resolution color VGA graphics down to two serial communication lines and a handful of graphics and text commands. This makes the ezVGA Serial Module a ready-to-use serial-controlled VGA video card that will deliver VGA functionality to your design. The nine easy-to-learn commands give any embedded controller or processor the ability to draw high-resolution color, VGA text, and graphics.

The ezVGA starts at **$24.95** and the ezVGA Serial Module is **$64.95**.

Multilabs
www.multilabs.net

M16C- The broadest platform with true code and pin compatibility

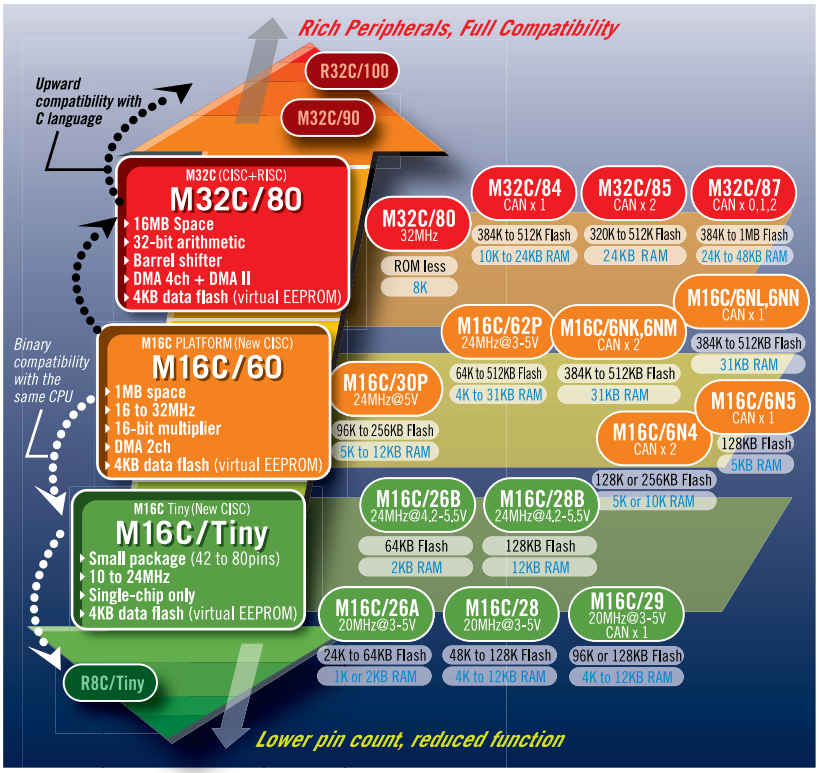
42 to 144pin, 24K to 1M Byte Flash & 1K to 48K Byte RAM

Renesas Technology

No.1* supplier of microcontrollers in the world

presents a wide range of M16C microcontrollers. M16C is the only fully code-compatible platform in the industry that addresses the entire 8-bit through 32-bit price/ performance application space.
Your application can range between 24K Bytes and 1M Bytes of code size, with between 42pins and 144pins of package size, while keeping the same code base and development tools.
The consistency and compatibility of the M16C Platform enables you to reduce your development time while still allowing the flexibility to adapt to changing system requirements.

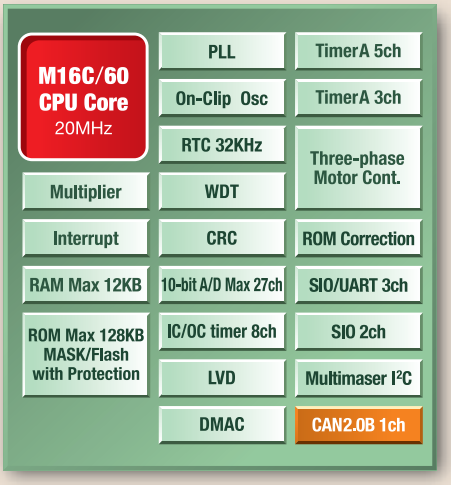
M16C Product Roadmap



*Source: Gartner Dataquest (April 2006) "2005 Worldwide Microcontroller Vendor Revenue" GJ06333

HOT Products | **M16C/29 Group**

M16C/29 Block Diagram



Top Reasons To Select M16C

- **Broad Platform** - Wide range of selection; 24KB to 1MB on-board Flash, 42pin to 144pin
- **Compatibility** - Pin and Code compatibility across the platform enables you to upgrade/downgrade MCU in same package without changing the board design and peripherals.
- **Powerful** - High speed interrupts (M32C: 0.61u sec in 32MHz) , optimized instructions for 1cycle operations, Hi-speed hardware multiplier.
- **Versatile** - Specialized on-chip peripherals; CAN, LIN, 3ph-PWM, DMA, etc
- **Efficient** - Multiple clock sources and multiple power saving modes
- **Quiet** - Built-in noise cancellation circuits provides excellent EMI/EMS characteristics that outperform IEEE standard spec
- **Easy** - Same Tool Chain for evaluation and development across the board
- **Reliable and secured** - Trusted flash and built-in fail safe features such as oscillation-stop detection circuit, protect registers, enhanced WDT, etc.

#1 MCU REACH FURTHER

Get Started Today -

Go online and register to be eligible for a FREE Starter Kit

www.america.renesas.com/ReachM16C/d



Visit us at ESC Silicon Valley

Embedded Systems Conference 2007 SILICON VALLEY

April 3-5, 2007

Booth #716 & #3011

Renesas Technology Corp.

Everywhere you imagine. RENESAS

NEW PRODUCT NEWS

RTOS SUPPORT FOR THE ARM RealView DEVELOPMENT KIT

embOS, the popular, small footprint RTOS, now supports the ARM RealView Microcontroller Development Kit (MDK) feature-rich development environment. This development gives customers a complete solution.

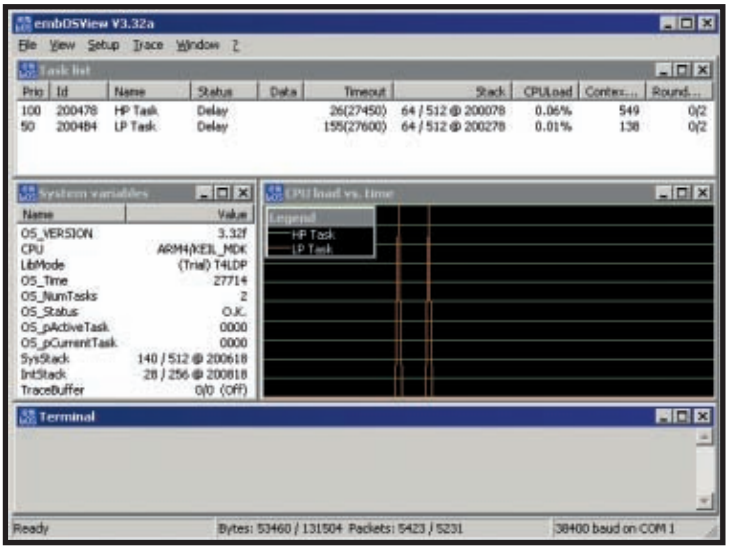
embOS, which is well known as a small and efficient real-time kernel, has become one of the most popular kernels for ARM processor-based microcontroller development over recent years.

The RealView MDK, which features the ARM RealView compilation tools and the Keil µVision 3 IDE, provides a complete development environment for users of standard microcontrollers based on the ARM7, ARM9, and the ARM Cortex-M3 families of processors.

embOS provides the benefits of a fully featured multitasking system by offering 100% identical API on all of its 8-, 16-, and 32-bit CPUs. Its small memory footprint also makes it suitable for single-chip applications. embOS also provides, at no additional costs, a viewer that allows you to look into the system from the PC (including profiling).

Note that embOS, which is available in object and source code form, comes with a single developer license with no project limitations. embOS is royalty-free and comes with six months of technical support and free updates.

The object code for the embOS-ARM7, ARM9, and KEIL MDK is available for **$3,100** and the source code is available for **$6,225**.



SEGGER Microcontroller Systems, LLC
www.segger.com



NEW PRODUCT NEWS

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

CONTROLLER WITH EASY-TO-USE SEQUENCING SOFTWARE

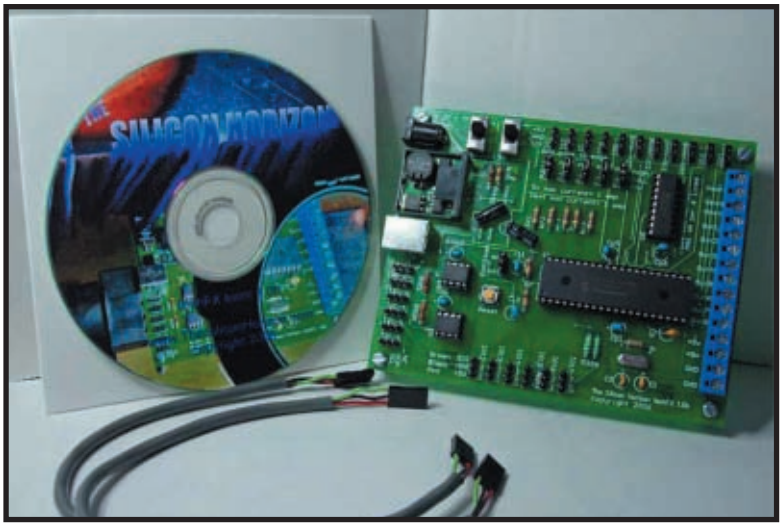
The new **techFX** controller is based on the Microchip Technology PIC18F4550 running at 48 MHz. It is feature filled enough to be used as a stand-alone controller board in most applications. Adding new functionality is as easy as connecting the new module and updating the configuration and commands. New modules include PWM, expanded I/O with a relay board, audio record/playback, and motor-control functionality.

Programming options are geared at both the veteran programmer and the customer who does not want to write one line of code. You can easily configure and create control sequences using the Windows-based controller software, or you can program the board using prewritten firmware code that already contains the backbone functions of USB, I²C, PWM, ADC, and many more functions. Use state-of-the-art tools such as Microchip's MPLAB IDE and MCC18 C compiler to quickly produce efficient code. Also, there are other options such as PICBASIC, HI-TECH C, and ASM. Then use the included bootloader to easily transfer your firmware to the board.

The board features full-speed USB 2.0 communications and eight high-current outputs through a Darlington array. Also available are 17 configurable I/O lines featuring up to eight 10-bit analog inputs, up to two PWM outputs, and digital I/O.

The techFX controller board costs around **$60** and includes the board, cables, and a CD containing the techFX tools.

The Silicon Horizon
www.thesiliconhorizon.com









ATir Keyboard Interface

Steven's well-designed ATtiny45-based ATir interface device is a convenient cross-platform solution to interface an IR remote control to type keyboard macros to a PC. The system features an infrared receiver/demodulator and a few discrete components.

At an early age, my son took a great interest in the "Thomas the Tank Engine" series. A good deal of his playtime was spent pushing those little wooden trains around the track. Part of the experience was to watch the videos, and as such, Dad often got invited to watch as well. "Little engines can do big things" is one of Thomas's favorite mottos. But what about little microcontrollers, can they do big things? They sure can.

My Atmel ATtiny45-based ATir interface device can emulate the function of a PC "AT" keyboard (see Photo 1 and Figure 1). With an IR remote control, the compact system (2" × 2" × 0.6") connects in series with an existing keyboard. If the portable device is already programmed, you can use the ATir without a keyboard (see Figure 2).

SYSTEM OVERVIEW

Small on pins does not necessarily mean small on performance. Take the Atmel ATtiny series. These tiny microcontrollers can do a lot more than act as a programmable 555 timer or implement some simple logic function. In this project, the microcontroller is used to decode and encode PS2 keyboard signals and perform data compression and storage in real time. It can also be used to decode signals from an IR remote, with a few external components to get the job done. If you're more of a software person and less of a hardware designer, this may be the project for you. I admit that Figure 1 is rather simple, but isn't there a certain elegance in simplicity? Another advantage of this simple design is its ease to implement on a budget.

The human interface to most PCs consists of a keyboard and a mouse. While this is convenient for many tasks conducted with a PC, some applications warrant remote operation. For example, when conducting a Microsoft Power Point slide presentation, it is nice to be able to advance the slides without having to operate the keyboard. For PCs dedicated to running multimedia applications in a home theater environment, it is easier to press the Play button on a remote control than to operate a keyboard. Some industrial systems use PCs running a custom software application that may only require a small set of keyboard commands to operate, while most of the time the keyboard remains unutilized, collecting dust. Those are just some of the applications that could benefit from the use of the ATir keyboard interface.

Using a keyboard, the ATir is programmed to respond to a key press on

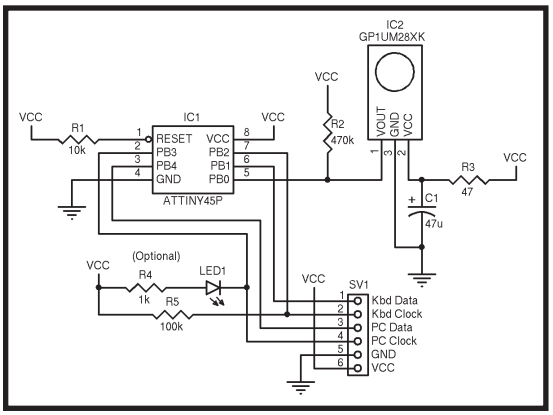


Figure 1—*As you can see, most of the passive components are for the Sharp IR receiver. Power comes from the PS2 port on the PC. No crystals or external clock components here. It uses the internal, calibrated RC oscillator with good results.*

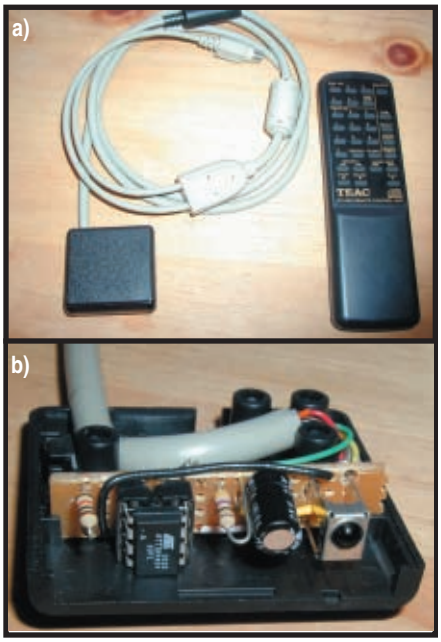


Photo 1a—*The ATir has an attached cord set salvaged from a combination keyboard-trackball device. The remote control is an inexpensive surplus unit, but the ATir is designed to work with other remote controls as well.* **b**—*Here is what it looks like inside. Not much to it, is there? The ATtiny45 is on the left, the IR receiver is on the right, and there are a couple passive components to glue it together. Even with the through-hole construction, there is no problem getting things to fit.*

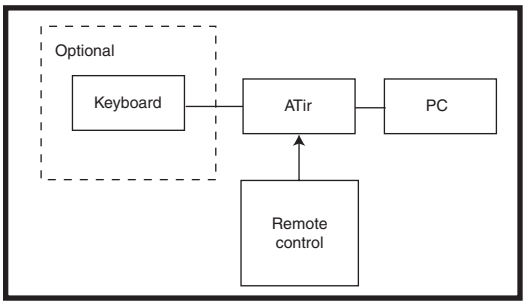


Figure 2—*The ATir connects a PC and a PS2-style keyboard. After the ATir is programmed, the keyboard can be removed, if desired. With no keyboard attached, the ATir takes care of initialization during the PC boot sequence. It is operated by an IR remote control.*

a remote control and output a block of keyboard data. For example, the ATir could be programmed to work with Power Point by entering Program mode, selecting a button on the remote control (say your remote has a "Play" button), pressing the F5 key, and saving the keyboard macro by pressing the same button on the remote. Now, with Power Point started, when you press the Play button on the remote, ATir sends the codes for the press and release of the F5 key and the Power Point presentation starts. To go to the first slide in the presentation, the keyboard shortcut 1 + Enter can be used. This keyboard macro could also be linked with another remote control push button.

Key sequences of up to 64 keys can be stored in an individual macro. These macros are stored in the EEPROM section of the microcontroller, so the data is retained after power is removed. After macros are defined, the ATir can be connected to a PC without a keyboard and it will perform the necessary handshaking during boot up to enable operation. A keyboard can also be connected to the ATir after the system boots and the ATir will relay information between the keyboard and PC.

KEYBOARD

The AT keyboard is configured with a five-pin DIN connector or a six-pin mini-DIN (commonly referred to as PS2) connector. The interface comprises four wires (i.e., 5 V ground, clock, and data). The clock and data lines implement a bidirectional synchronous serial bus. The bidirectional nature of the bus allows the PC to send select messages to the keyboard such as, retransmit the last byte, or turn on the Caps Lock LED, as well as the normal traffic from the keyboard to the PC.

Key press information is transmitted in the form of a scan code. For example, the A key sends a code of `0x1C` when pressed and `0xF0 0x1C` when released. The key press and key release are coded separately to allow multiple keys to be pressed simultaneously (such as when using the Shift key to type a capital letter). The format for a scan code is an 11-bit frame, consisting of a start bit, 8-bit code, odd parity bit, and stop bit. The frame-timing diagram for keyboard to PC (host) communication is shown in Figure 3a. PC-to-keyboard communications is initiated by the PC by placing the bus in a "request to send condition," which signals the keyboard to clock in the command. This communication is shown in Figure 3b.

IR REMOTE OPERATION

The infrared remote used for this project transmits a 32-bit packet consisting of a 16-bit address field, an 8-bit command field, and an 8-bit command complemented field for error detection. The IR signal is modulated by the IR remote at approximately 38 kHz. Other modulation frequency receivers are available in the Sharp GP1UM27XK series from 36 to 40 kHz. With no signal, the device outputs a high value and it will drive the input low with the presence of a modulated IR signal. For the remote control, a logical 1 is represented by a short pulse of approximately 540 µs. A logical 0 is represented by a long pulse of 1,640 µs, with a 570-µs delay between pulses. Most of the discrete components used in this project comprise the low-pass filter recommended in the Sharp GP1UM27XK datasheet.

The ATir was designed to support a variety of remote controls by reading remote control commands at run time and not relying on particular commands from a particular remote. The remote used was a surplus unit from All Electronics (a TEAC model RC505), but other remote controls could also be made to work. Although the 16-bit address field is decoded by the ATir, it is ignored so other remotes with different ID values can be used with no modification to the code. A universal remote control can be configured to work with the ATir, as well as extra remote controls that you may have that are not being used (that modulate at the same frequency and use a common 32-bit frame format).

ATir HARDWARE

The ATir is used to interface an infrared remote control to a PS2 keyboard port, where a key on the remote corresponds to a programmable keyboard macro. The device has male and female mini-DIN connectors that are intended to be connected inline with a PS2 keyboard, but using a keyboard is optional. I was able to harvest a cable from a broken keyboard/trackball combination keyboard and change one of the six-pin male mini-DIN connectors to a female connector. When a keyboard is attached to the device, the device relays scan codes and command information to and from the PC and the attached keyboard. If a keyboard is not connected, the device will perform the required handshaking with the PC simulating the

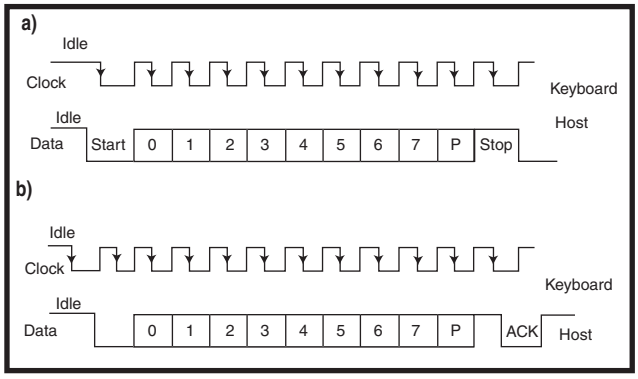


Figure 3a—*As you may expect, the majority of communication is from the keyboard to the PC. The keyboard generates the clock signal and data is sent in an 11-bit frame.* **b**—*Yes, the PC can actually send data to the keyboard (such as turning on the Caps Lock LED). The keyboard generates the clock signal regardless of the direction of dataflow.*

response of a keyboard. Without this capability, the PC BIOS would determine that a keyboard is not connected and either present with a keyboard error message and halt or turn off the port to save power (with the response being dependent upon how the BIOS is configured).

The device hardware, as assembled, includes an IR-13 case from All Electronics, an ATtiny45 microcontroller, a Sharp infrared receiver/demodulator and a few discreet components consisting of low- pass filter components and pull-up resistors. Originally, the intention was to have a circuit board manufactured, but in the end, the circuit was constructed on a 0.1" center-perforated board. The ATtiny45 microcontroller IC1 is an eight-pin device that contains 4 KB of flash memory, 256 bytes of SRAM, and 256 bytes of EEPROM. The microcontroller also contains two 8-bit timer/counters, a universal serial interface, and a 10-bit successive approximation ADC. For this project, I used the ATtiny45 because the ATtiny85 was not yet available. The ATtiny85 has twice as much memory (flash memory, SRAM, and EEPROM) in the same pin-compatible package, allowing for twice as much keyboard macro data to be saved, as well as an enhanced user interface. This will be discussed more in the software section.

The keyboard clock line is connected to the external interrupt pin and the interrupt is configured for falling-edge detection. The output of the IR detector IC2 is sensed using the pin-change interrupt with a mask configured to sense only the IR detector output. To save space and components, the internal pull-up feature is used instead of the external pull-up resistors for the required PC clock and data lines. A high-resistance pull-up R5 is used on the keyboard clock line to allow operation with a keyboard disconnected while being able to sense and operate a keyboard when connected. Resistors R4 and LED1 were used during development. They give a visual indication whenever the ATir is transmitting data to the PC. This occurs when a key is pressed on an attached keyboard or when an IR command that is linked to a valid keyboard macro is received and sent to the PC. Indicator connection is possible because the ATir always generates the clock signal, and the ATtiny45 has no trouble driving LEDs directly from an I/O pin.

Ultimately, I did not include the LED because of a peculiarity in the PS2-to-USB adapter, which I purchased to extend the functionality of the ATir to PCs, with no PS2 keyboard/mouse ports. It turns out that the adapter periodically sends an invalid AT scan code that the keyboard responds to per the AT keyboard protocol. Maybe this is done to detect if the keyboard has been removed, and I don't know if all the adapters operate in this manner. The result of this periodic handshake (about every second or so) would cause the LED to constantly flicker while the adapter transmitted its response. Even engineers grow weary of constantly flickering LEDs at some point.

The system doesn't have too much hardware, so it's fairly easy to build. The internal, calibrated eight-MHz clock was used as a clock source for the ATir. This source proved to be surprisingly accurate, as the timing requirements for the AT keyboard data communication and IR pulse length timing require a consistent and accurate system clock.

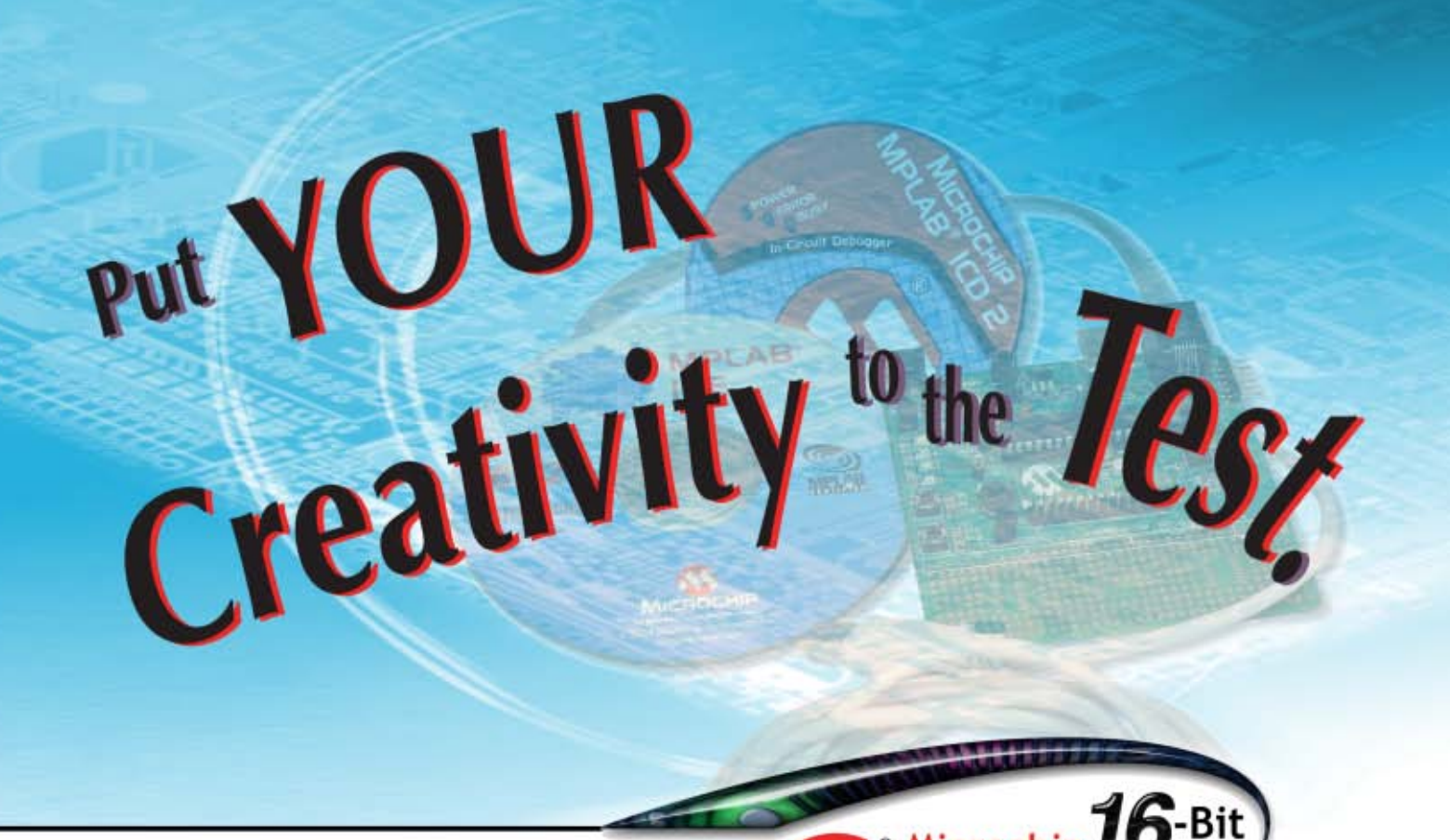
Most of the digital resources of the ATtiny45 are used in this project. Timer0 is used to time the pulses from the IR receiver. Timer1 is used to implement a timeout condition during transmit and receive to ensure that synchronous communications remain synchronized. The EEPROM section is used to store IR commands and the

Listing 1—*How can a keyboard create a user interface? Just "type" it out. This algorithm takes an ASCII character input and converts it into an appropriate key down, key release, and shift key press (if necessary) scan code stream. Anyone for a speed-typing contest?*

```
//This function "types" the character argument via output of
//keypress scancodes
void kbd_putchar(char ch){
   uint8_t index;
   uint8_t keyCode;
   bool shift = false;
   switch (ch){
   case ('\n'):
   keyPress(KEY_ENTER);
   break;
   case ('\t'):
   keyPress(KEY_TAB);
   break;
   default:

   index=ch-32;   //subtract 32 from ascii value to obtain index
   if (index>63){ //lower case letters and supported symbols
                  //greater than 63
      index-=32;  // are equiv to value - 32 but with shift inverted
      shift = true;   //after xor, shift is inverted
   }
   keyCode=pgm_read_byte(codeTable + index);
   if bit_test(keyCode,7){
   shift^=true;       //set shift or invert if already 1
   bit_clear(keyCode,7);
   }
   if (shift){ //this section presses the shift key for every character
   keyDown(KEY_SHIFT);
   keyPress(keyCode);
   keyRelease(KEY_SHIFT);
   }
   else
   keyPress(keyCode); //non shift keypress

   break;
   }//end switch
}
```



Circuit Cellar magazine and Microchip are pleased to bring you the Microchip 16-bit Embedded Control Design Contest.

Microchip 16-Bit Embedded Control Design Contest

This is the perfect opportunity to learn about Microchip's 16-bit Microcontroller and Digital Signal Controller (DSC) families. These powerful workhorses offer a broad range of products while preserving the compatibility that helps save you time and money on subsequent designs. Put your creativity to the test. Your project could win you a share of $15,000 in cash prizes and give you the international recognition you deserve.

Learn about unique contest sample opportunities, special offers from Microchip, and how you can be the next Circuit Cellar design contest success story.

For details, visit www.circuitcellar.com/microchip2007

CIRCUIT CELLAR®

MICROCHIP
The Embedded Control Solutions Company

associated macro data. Program memory utilization is a few bytes short of the ATtiny45's 4-KB capacity. All of the I/O pins are used. The RESET pin has the capability of being reconfigured as a general-purpose I/O, but this feature was not implemented because reprogramming the chip would then require a high-voltage programmer versus the inexpensive parallel port programmer that was used.

The microcontroller source code was developed in C using AVR GCC in the AVR Studio 4 development environment. Working on a budget, I thoroughly enjoyed working with the free AVR GCC compiler that integrates nicely with AVR Studio. Initially, code was written and tested using the built-in simulation tools in AVR Studio for a 40-pin ATmega32 as the target.

Next, the application was programmed into the ATmega32 for in-circuit debugging. This allowed for easy programming and debugging by using a JTAG ICE and a built-in UART to send scan code information to and from the device to a PC running a terminal program. The ATmega32 also had plenty of spare I/O lines to run an LCD and output signals to capture events on an oscilloscope.

After the application was running on the ATmega32, some minor architectural changes allowed the application to be ported on the ATtiny45 using the same compiler and development environment. Another debugging option is debugWIRE, the one-wire-debugging feature of the ATtiny45, but this requires a JTAGICE mkII, which didn't fit into the budget.

The original plan was to port the application to the ATtiny85, which is physically the same chip with double the memory (flash memory, SRAM, and EEPROM) of the ATtiny45. Unfortunately, this chip was not available during development, requiring a lot of cutting (and almost hammering) to get the application to fit. With an ATtiny85, the user interface and programming functionality can be improved significantly. I have a few suggestions.

Function argument validation can be added. When entering Programming mode, show a listing of macros that have already been programmed. Indi-

Listing 2—*This algorithm reads the stream of scan codes and extracts the essentials. The ATtiny45 has a 256-byte EEPROM to store all macro data. Conservation is "key."*

```
void encode(uint8_t scancode){
static bool ext = false;  //extended key flag
static bool rel = false;  //key release flag
static bool lwc = false; //last (input) was a code byte (not ext or rel)

   static uint8_t lastcode = 0;
       switch (scancode){
       case 0xE0:
           if (lastcode && !ext){
               addKeyDown(lastcode);
               lastcode = 0;
           }
           ext = true;
           lwc = false;
           break;
       case 0xF0:
           rel = true;
           lwc = false;
           break;
       default:
           if (scancode==lastcode){
               if(rel){ //if release, its a keypress
                      rel=false;
                      if(ext){
                            ext = false;
                            addKeyPressExt(lastcode);
                      }
                      else{
                            addKeyPress(lastcode);
                      }
                      lastcode = 0;
               }
               else{
                      if(ext && lwc){
                            addKeyDownExt(lastcode);
                            ext = false;
                      }
               }
           }
           else{//scancode != lastcode
               if(lastcode){    //a key was pressed pending
                      if(rel){
                            rel=false;
                            f(ext){
                                  ext = false;
                                  addKeyDownExt(lastcode);
                                  addKeyReleaseExt(scancode);
                            }
                            else{
                                  addKeyDown(lastcode);
                                  addKeyRelease(scancode);
                            }
                            lastcode=0;
                      }
                      else{
                            if(ext){
                                  addKeyDownExt(lastcode);

                                  if(lwc)
                                        ext = false;
                      }
                      else{
                                  addKeyDown(lastcode);
                      }
               lastcode=scancode;
               }
```

(Continued)

Silicon Labs' ToolStick: Plug 'n Debug



Fully Contained MCU Development System

Not only did Silicon Labs introduce the worlds' first USB ToolStick evaluation platform, we continued the evolution with a family of interchangeable ToolStick modules including base adapters, daughtercards and debug adapters. Now, software and hardware MCU development is easier than ever. Using only a PC with a USB port, designers can fully experience the software development environment in conjunction with the MCU on-chip debugging hardware that allows full, non-intrusive access to the target CPU, peripherals and memory. The software development environment consists of the Integrated Development Environment (IDE), editor, debugger, Flash programmer and a demonstration version of the Keil compiler.

Description	Price
ToolStick Starter Kit (includes C8051F330 Daughter Card)	$24.99
ToolStick Evaluation Kit	$10.99
ToolStick Base Adapter	$17.50
ToolStick C8051F330 Daughter Card	$9.90
ToolStick C8051F360 Daughter Card	$9.90
ToolStick C8051F530 Daughter Card	$9.90
ToolStick Debug Adapter	$8.90

Buy Online
www.silabs.com



Complete Evaluation System for $10.99

- Development Tools
- Source Code Examples
- Documentation
- 2K Limited C Compiler

Product details: www.silabs.com/ToolStick

SILICON LABS

cate the currently used and available space remaining for programming macro data. Allow the last macro programmed to be deleted (easy), or any macro programmed (more difficult).

So, how do you make a user interface with a device that emulates a keyboard? Simply start your favorite text editor and let the keyboard type the interface text. To simplify this process, the ATir implements a `putchar` and `puts` function that includes all letters, numbers, and many of the special characters found on the keyboard (ASCII codes 32–125) using a 64-byte look-up table. Keyboard scan codes don't follow a logical encoding scheme that could be derived numerically; therefore, characters are translated into scan codes via the look-up table. Both uppercase and lowercase characters can be sent, because the algorithm inserts the Shift key into the scan code stream when a capital letter or a special character like those printed on the number keys (e.g., !@#$%^) is required. The scan codes for keys associated with printable characters can be expressed with 7 bits. The eighth bit is encoded in the look-up table value to indicate if the Shift key should be pressed to output the character.

ASCII codes below 33 are not printable, so no table entries are necessary for them. For ASCII codes above 96, the table is offset by 32 and the shift bit state is inverted. By using this scheme, the lowercase characters beginning at ASCII code 97 do not have to be encoded into the look-up table. The table was created in Microsoft Excel, which assisted in converting numbering systems and calculating for the shift bit and formatting the data in columns that could then be copied and pasted directly into the header file (see Listing 1).

The application places some real-time signal processing demands on the microcontroller. Reception of data from the keyboard must be processed and retransmitted to the PC. Keyboard data is processed by a filtering and compression algorithm when recording macros to conserve EEPROM memory. A key press and release sequence causes the keyboard to transmit three or five scan code frames comprising 11 bits per frame. This sequence is represented by storing 1 byte in EEPROM. In addition, if a character key is held down in an application (such as a word processor), the character repeats until it is released. This is because the keyboard resends the key down scan code at a configurable periodic rate, which is referred to as the typematic rate. This typematic data is filtered and therefore does not contribute to storage requirements. Listing 2 is the compression algorithm.

Listing 2—*Continued*

```
            }
            else{       //key not pressed pending
                        if(rel){
                              rel = false;
                              if(ext){
                                    ext = false;
                                    addKeyReleaseExt(scancode);
                              }
                              else{
                                    addKeyRelease(scancode);
                              }
                        lastcode=0;

                        }
                        else{

                        lastcode=scancode;
                        }
            }
        }
    lwc=true;
    }//end switch

}
```

OPERATION

To program keyboard macros using the ATir, simply plug the male connector into the PS2 keyboard port of the PC, plug your keyboard into the female connector of the ATir, and boot your computer. Open your favorite text editor. For Windows operating systems, Notepad is sufficient. To enter Programming mode, a key sequence is entered on the keyboard. To prevent the ATir from going into Program mode during normal typing tasks, I chose a keyboard sequence that is somewhat abstract. Type the following sequence: hold down the right Shift key, press and release the left Shift key, and then release the right Shift key. Hopefully, you have never used that sequence before with normal computing tasks. Now, as if by some extremely fast magic speed typist, text should be scrolling onto the screen prompting you to press a button on the remote. If the key sequence above is reentered at this stage, the EEPROM is cleared.

To program a macro, point your compatible remote control at ATir and press a button. If decoded successfully, the command byte is displayed and you are prompted to enter a macro and press the same button to record. Type in your macro. When you're finished, press the same button on the remote control to confirm the macro and record it. If a macro does not exist for the command selected, a prompt will indicate that the macro was recorded successfully. When you press the Remote Control button, the keyboard macro sequence is sent to the PC. Reenter Program mode to set up additional macros for other keys on the remote. Listing 3 is a programming session.

NICE SOLUTION

The ATir keyboard interface offers a convenient cross-platform solution to interface an IR remote control to type keyboard macros to a PC. The interface plugs into a PS2 keyboard port on the PC and accepts commands from an infrared remote. The device can be configured to operate with common remote controls with little or no change in code. For those of you with laptops or

Listing 3—*This is an example of a programming session creating a "Hello World" macro. The ATir "types" the first line after entering Programming mode. I press a button on the remote that is decoded as "EF." The ATir prompts me to type in a macro. I type "Hello World" <Enter> and then press the same button on the remote. The ATir responds "OK." I press the Programmed button on the remote twice and "Hello World" is printed on the screen twice.*

```
Press button on remote to program

Command = EF
Type in macro now
Press the same button on remote to accept
Hello World

Command = EF
IR command match
Macro write OK
Hello World
Hello World
```

new computers without PS2 ports, do not despair. USB-to-PS2 adapters are inexpensive and allow the ATir to be used within USB environments.

Another option would be to bypass the adapter and design the ATir with a USB connection. Directly implementing a USB interface is not trivial. Atmel has a family of microcontrollers with hardware support of the USB protocol that would be more appropriate for that kind of project (compared to the ATtiny45 used in this project). Of course, if you want to connect your USB device to an older PC with only PS2 ports, you'll need an adapter. The ATir worked well with the USB adapter, which is nice for Power Point presentations from your laptop on the road. The adapters are readily available on the 'Net for less than $10, including shipping. Just don't forget to program it before you leave, unless you have time and access to a PS2 keyboard.

Other remote controls for PCs are available, but the ones I have seen require using the supplied remote and installing drivers. With the ATir, you can recycle that old working remote control, even after the device it controlled has passed away. The system will work with any PC operating system that recognizes a keyboard. As constructed, it costs about $10.

Steven Savage holds a Computer Engineering degree from the University of Akron. He is currently pursuing a Master's degree in Electrical Engineering (special thanks to the faculty and staff in the department of Electrical and Computer Engineering at the University of Akron, especially Dr. Grover). Steven works for Daimler Chrysler as a welder repair technician. He is also working on a technology startup business called Vigilant Technology. He enjoys playing trains with his son, archery, working with the media team at his church, and adding blinking LEDs to fishing lures. You can reach him at stevensavage@nls.net.

PROJECT FILES

To download code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/201.

RESOURCES

Atmel Corp., "ATtiny datasheet, 2586-AVR," 2586J-AVR, 2006, www.atmel.com/dyn/resources/prod_documents/doc2586.pdf.

C. Peacock, "Beyond Logic: Interfacing the AT Keyboard," 2005, www.beyondlogic.org/keyboard/keybrd.htm.

Sharp Corp., "Sharp GP1UM27XK series datasheet," ED-01052A GP1Um27XK series, 2001, http:// rocky.digikey.com/WebLib/Sharp/Web%20Data/GP1UM27XK_Series.pdf.

SOURCES

ATtiny45 Microcontroller
Atmel Corp.
www.atmel.com

GP1UM27XK IR Remote receiver unit
Sharp Corp.
www.sharpusa.com



Java-Based EarthBox Watering System

Tom was tired of watering his plants by hand, so he built a Java-based 1-Wire distributed watering system. The expandable system can water six EarthBoxes.

Some time ago, I became interested in EarthBoxes. "EarthBox" is the trade name for a commercially available gardening box. The boxes have been growing in popularity. They are ideal for folks like me, who have limited access to sunlight. (I live at the edge of a forest.) They are also ideal for people who want the conveniences of a productive garden without having to devote too much time and real estate (garden area) to one.

The boxes are self-contained and engineered with soil containers, an internal water reservoir, and fertilizer reserves to yield hearty vegetable gardens from a modest scattering of 2" seedlings. Newer EarthBox designs are equipped with casters for mobility and an integral staking system (see Figure 1).

In this article, I will describe how I designed and built a Java-based 1-Wire distributed watering system for an EarthBox. The compact system is shown in Photo 1.

EARTHBOXES

When I started using EarthBoxes, I was amazed at the number of tomatoes and eggplants harvested from just four plants (two plants per box). To keep the plants healthy, the boxes require regular watering to sustain their internal reservoir levels, because the plants steadily soak up water. Watering is pretty infrequent to start with when the plant is small. However, as it matures and bears vegetables, watering is extremely important. Water levels must be maintained (at least daily) to yield a reasonably sized fruit. The internal water reservoir, which is close to the soil just above it, allows the soil under the plant to access water as needed. To facilitate the watering process, each EarthBox has a watering tube that empties into the internal reservoir. As the internal reservoir fills to capacity, a seepage hole in the side of the box releases water to ensure that you do not overwater the plant. This is good engineering.

When I started growing plants in EarthBoxes, I would slip a garden hose into the watering tube of each box, let it fill, and then note when the seepage hole released water. I would then move on to the next box and repeat the process. I eventually had six boxes. (I do like my fresh vegetables!) Watering all six boxes every day can be a chore. That's why I developed an automatic distributed watering system that sustains the reservoir levels with a programmable watering cycle. I also wanted to be able to monitor (on a per-box basis) the amount of time the system spends watering each box. I thought the monitor would give me a good indication of whether the EarthBoxes were functioning and the plants were growing. The system needed to be distributed to allow me to judiciously situate the boxes where they could benefit from maximum sun exposure and not "crowd up" in any one area around the house. I envisioned a totally hands-off gardening system that would require lit-

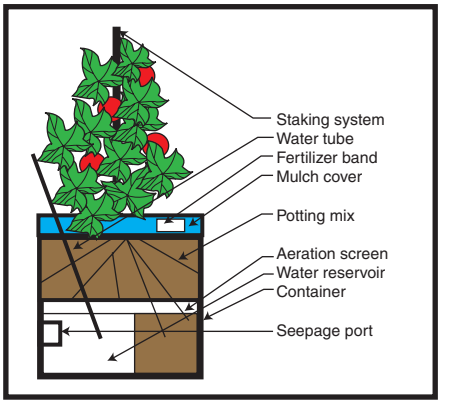


Figure 1—*A cut-away illustration of an EarthBox.*



Photo 1—*An LCD and a keypad are in the center. The Javelin Stamp is on the right. The RJ-45 connections are in the rear of the box. A dual-color LED flashes to indicate the system's overall status. Green indicates proper operation (otherwise, it flashes red).*

tle more than planting, fertilizing, and harvesting.

Luckily, the watering technology exists; it is most prevalent in lawn irrigation systems. Garden hoses are inexpensive, and water distribution is easily accomplished with lengths of a hose and garden hose manifold distribution fixtures. Watering control can be accomplished with inexpensive lawn watering solenoid-controlled valves. The valves can connect to garden hoses with PVC pipe and the proper hose fittings. To control these valves and monitor water levels, I used Maxim Integrated Products's 1-Wire technology. I had recently completed a larger project using the technology and I was impressed by its simplicity and reliability. I knew that using 1-Wire would allow me to uniquely address, monitor, and control all of the valve controls and watering levels for each remote box with a single location. I also knew that 1-Wire technology would support several EarthBoxes. Given the right 1-Wire driver, cable type, and terminations, total cable lengths of 300 m are possible. Such a setup can accommodate more than 100 1-Wire devices (that's about 50 EarthBoxes in my setup).



Photo 2—*The remote box supplies water to each EarthBox via black hoses. It contains an internal waterproof box for the electronics, as well as a PVC manifold and valves. The white water sensor wires are coupled to the hose and feed back to the remote box.*

The other key element was the microcontroller. I wanted to use Java language. Again, I had just completed a project with Java and I was impressed with its object-oriented features, its reuse capability, and its extensibility to larger platforms. Microcontrollers are available in all shapes and sizes, but there are limitations on what is easily available for embedded Java. I chose the Parallax Javelin Stamp, which is a flash memory reprogrammable SBC (about the size of a postage stamp) with 32 KB of RAM/program memory and support for a host of virtual peripherals: UART, timer, ADC, DAC, ports, and buttons.

The Javelin is programmable in a Java language subset using the Parallax IDE. It has debug capabilities and Java library classes to easily deal with 1-Wire and embedded Java applications in general.

SYSTEM OVERVIEW

Each EarthBox can be a maximum of about 100′ from the Javelin-based controller box, which contains the Javelin controller, the 1-Wire interface, a 1-Wire distribution hub, system power, a time-of-day chip, a watchdog timer, a solid-state relay for solenoid power on/off, an LCD with a keypad for operator detail status/control, and a dual-color LED for general status indication (see Figure 2). The entire assembly is housed in a clear box (see Photo 1). The box is not waterproof, but I keep it outdoors, under my porch, for easier connection to the remote EarthBoxes.

The system also includes remote watering boxes (see Photo 2). Each remote box is connected to the controller box for control/status reporting and power via a modified RJ-45 cable connector scheme. The same type of clear box that is used for the controller is reused for the remote box (see Photo 3). In the larger remote box, a small plastic waterproof box is incorporated for

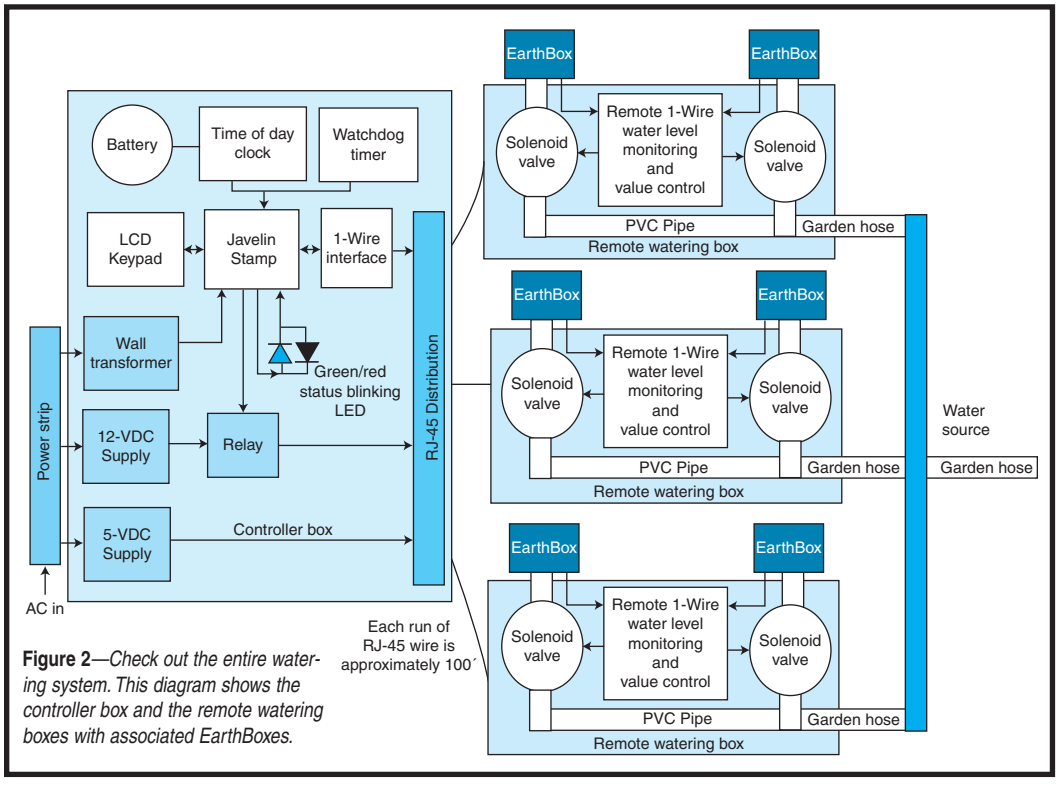


Figure 2—*Check out the entire watering system. This diagram shows the controller box and the remote watering boxes with associated EarthBoxes.*

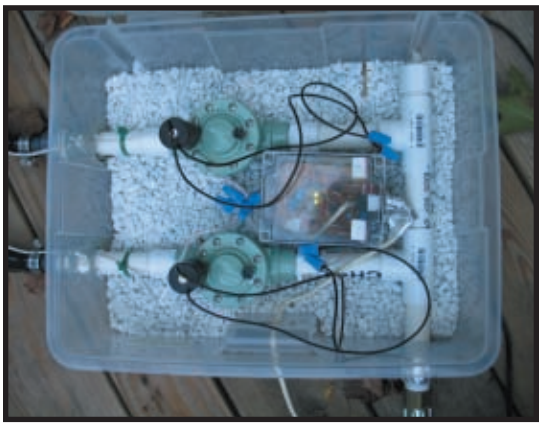


Photo 3—*The placement of the remote box's major internal components is clearly depicted. I opened the remote box's cover so you can see the location of the remote electronics box, the hose, the valves, the PVC manifold, and the feeder connections. What do you think? You can also see the sensor wires within the remote box.*

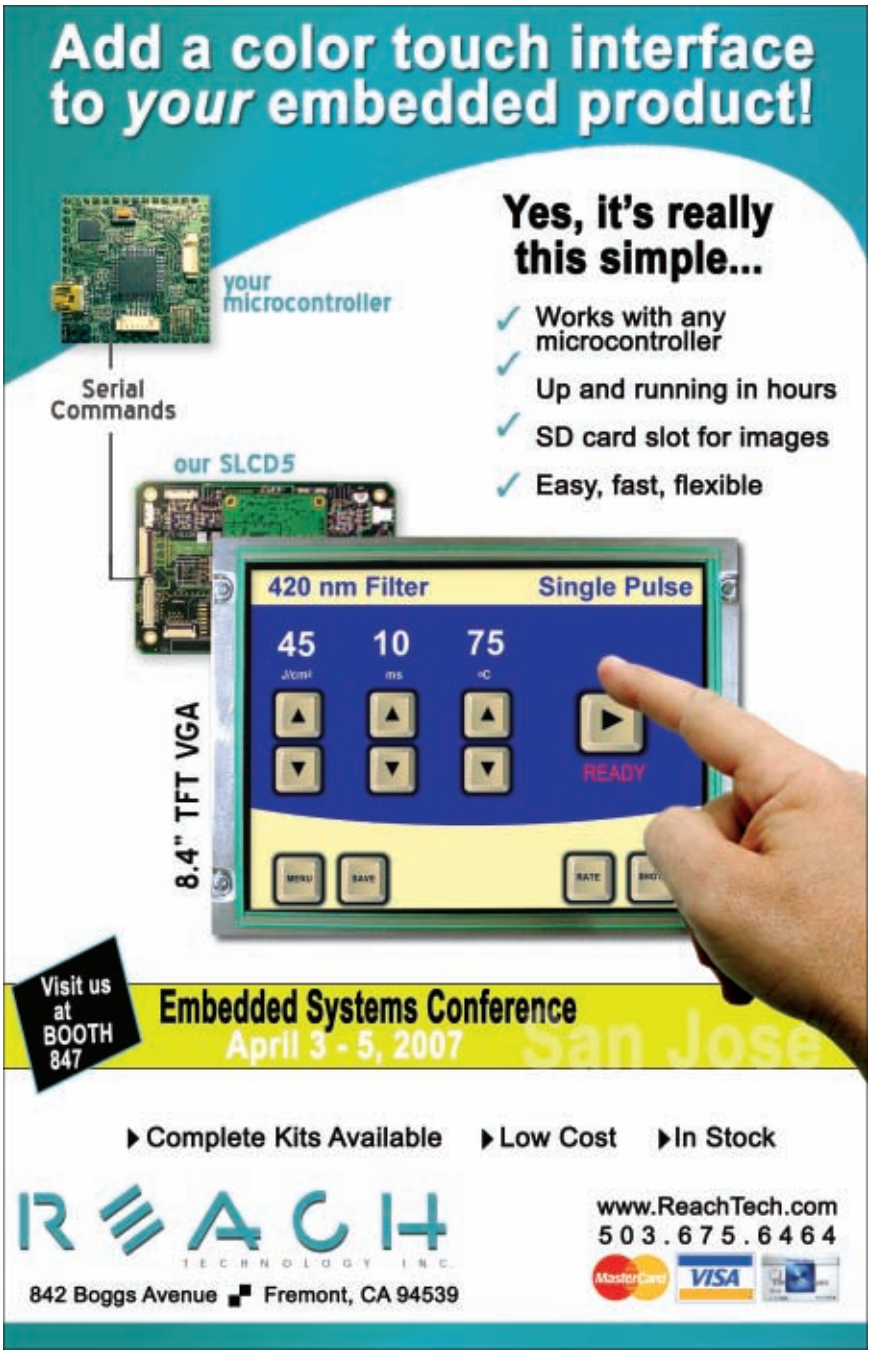
the 1-Wire valve control and water-sensing electronics. The solenoid irrigation valves and PVC manifold construction are also in the larger box. For each manifold, a garden hose must be connected for the water supply. I mounted a 0.5″ garden hose adapter to the water entry for each manifold.

The solenoid relay I chose was the 54048 0.75″ in-line valve from Lawn Genie, which is available at most hardware stores. To weigh down the remote box, I used crushed white rock (see Photo 3).

The controller monitors the water level in each EarthBox. With this data, it also supervises the watering process for each EarthBox through its solenoid water valve. Each valve and water sensor is uniquely addressed using 1-Wire. The watering scheme can be implemented automatically or manually; it is just a matter of software. I used both automatic and manual schemes. In Automatic mode, you can set the frequency of the watering cycle (from 1 h up to every 16 h) and the maximum valve/watering on time (from 1 min. to up to 20 min.). The maximum on time is a preventive measure to ensure that each valve does not stay on too long to achieve the desired water level (especially if water is shut off or if it is not connected to the remote box). The controller logs the accumulated watering times for each EarthBox and notes if a maximum on time has been exceeded. If a maximum on time has been exceeded, the controller will bypass this valve for the next watering cycle and periodically flash the dual-color LED (red). Otherwise, if all conditions are satisfied, the LED will blink green. You can also change the scheme to perform various other tasks because I designed it to be inherently flexible and expandable.

HARDWARE OVERVIEW

Power comes from a number of sources: a wall transformer to the Stamp, a 5-VDC power supply to all remote box electronics, and finally a 12-VDC power source to the valves. Individual returns for the 5 and 12 VDC were required to ensure noise-free operation. The 12 VDC is switched using a Kytech Electronics KF0604D KYOTTO solid-state relay. I realized



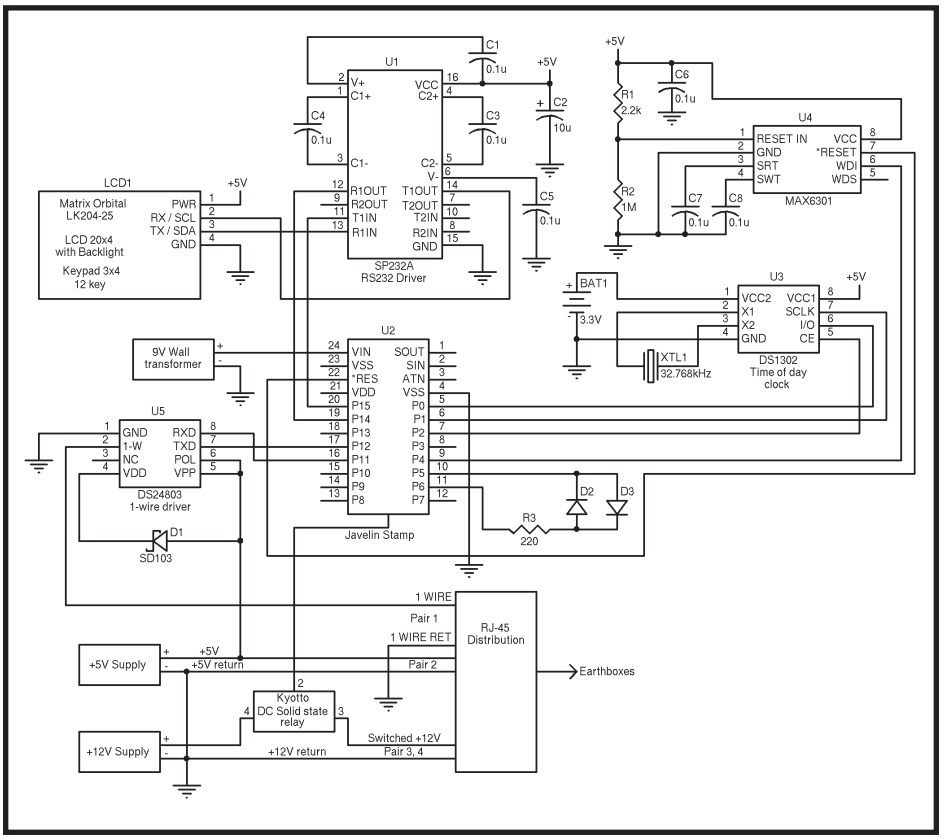


Figure 3—*The Javelin Stamp is connected to all of the peripherals. The RJ-45 cables exit the box through the RJ-45 distribution to the associated remotes and their EarthBoxes.*

that 12 VDC should only be applied once it has been decided to switch the solenoid relay. Note that 12 V at 0.5 mA is sufficient.

The wall transformer (9 VDC) is used with a Stamp prototype board and its internal regulator (National Semiconductor's LM2940) regulates the 5-VDC required by the Stamp. Because of the remote box's 5 VDC requirement (in comparison to what the Stamp prototype board could deliver), I used a separate 5-VDC supply. Both the 5 VDC and the 12 VDC of power are distributed to the EarthBoxes via pair 2 and pairs 3 and 4 of the RJ-45 connection respectively. Two pairs are assigned to the 12 V because of the larger current requirement. In addition, all ground returns are separated and connected to a "Mecca ground" in the controller box to minimize noise. The remaining RJ-45 pair is connected to the 1-Wire electronics.

The Javelin Stamp reports on the system with a blinking red/green LED. More detailed status and con-

WARNING

There are literally thousands of pages of innovative embedded systems applications crammed into one venue: www.circuitcellar.com. The expanded applications section we've created over the years includes source code, schematics, and design insight from your peers. It continues to grow by leaps and bounds, delivering an amazing online supplement to Circuit Cellar's print magazine.

It's all yours to view for free. Start with the latest project postings from the Atmel AVR 2006 Design Contest. With so many applications from so many designers around the world, you're bound to find a ton of useful tips and techniques. Check back often for the latest additions.

Start at www.circuitcellar.com/contests/

Contents Under PRESSURE



trol information is available via the Matrix Orbital LK204-25 LCD and its associated keypad. To support this LCD/keypad interface, an RS-232 interface to the Javelin Stamp's virtual peripheral UART is required. This interface is accomplished with the use of a Sipex SP232ACP RS-232 driver chip. The controller maintains the time of day, its watering cycles, and the elapsed time using the Maxim Integrated Products DS1302 TOD chip. The chip is backed up with a 3.3-V lithium cell battery. Again, the interface is well supported. The Javelin Stamp library can incorporate a DS1302 object for application reuse.

Finally, I used Maxim Integrated Products's 6301watchdog timer chip. Watchdogs are an important part for ensuring reliability within embedded designs by providing an external (non-software) reset to the controller on a periodic basis. If the controller is fully functional and operating within its code sequence, a provision is made within each code segment to provide a "reset" to this watchdog timer prior to

Listing 1—*Within the MAINTESTER class is a continuous-loop state machine. The code for one state within the loop is shown. Each individual valve and its associated water sensor represents a state. The system exits a state and goes to the next state once all of the conditions are satisfied relative to the valve operation and water level.*

```
switch (state) {
//start of automatic process
case 1:
     address =switch1;
     valve_on = valve_state1;
  //Test 1-if valve in error condition just bypass
  if (valve1_error){ state++; break; }
     water_ok =read_DS2406();
  //Test 2-if water ok and valve is not on just move on to next state
     if (water_ok && !valve_on){state++;break;}
     //Test 3-if valve isn't on and water is low, turn on valve start time
     //return to same stat
     if (!valve_on&& !water_ok)     {
        on_valve(1);
        captureStartime();
        t_run =0;    //start timer
        //display valve on-stay in same state
        d.valve_Status (1,valve1_ontime,true,valve1_error);
        d.outTimetoDisplay(t.readTime(true));  break; }
     //Test 4 -if valve is on and water is ok -turn off valve
     // and display time or error
      if (valve_on && water_ok) {
       off_valve(1);
        captureStoptime();
           t_run=  time_accum(0) ; //t_run =diference of stop-start\
           valve1_ontime=  time_accum(valve1_ontime) ;
           if ( t_run> delta ) valve1_error =true;
           d.valve_Status (1,valve1_ontime,false,valve1_error);
           d.outTimetoDisplay(t.readTime(true));
           t_run =0;
           state++;
           break;
     }
```



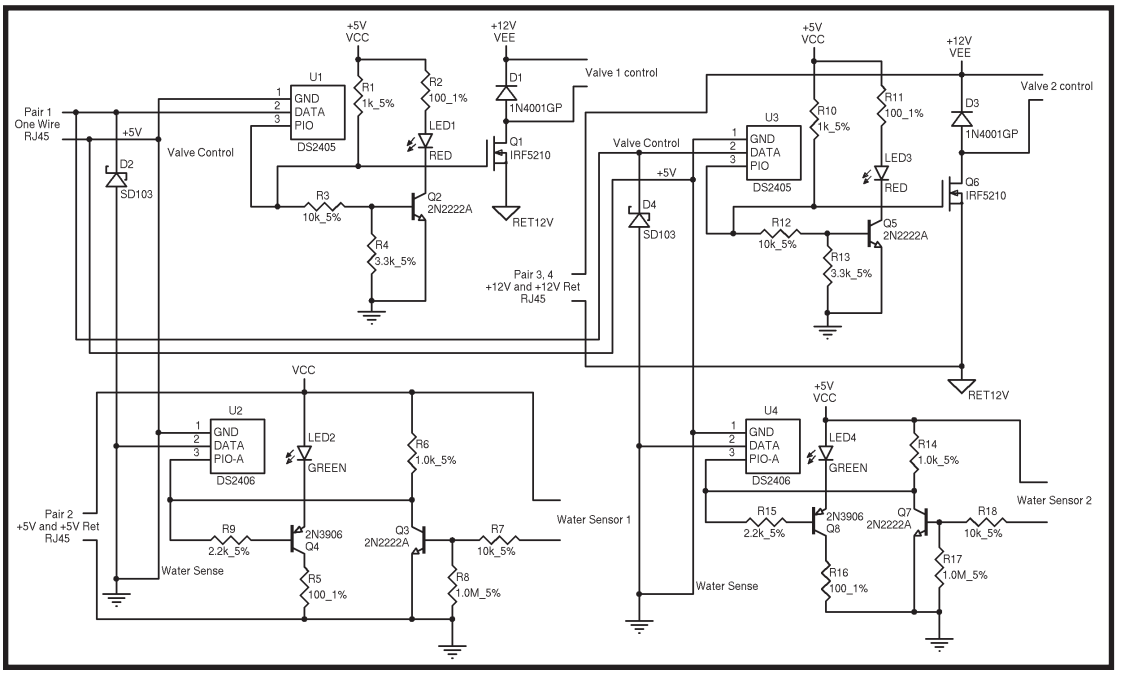



Figure 4—*The electronics are duplicated for each EarthBox, water control, and sensor. The entire box connects to a single 1-Wire interface. The DS2405 1-Wire device is used for valve control, while the DS2406 1-Wire device is used for water sensing. A red LED is connected to each valve control to indicate activity. A green LED is connected to each water sensor.*

its own timeout (whereby it sends a reset to the controller). Watchdogs are generally recognized in the industry for ensuring the reliable operation of an embedded controller. In this case, a 6-s watchdog timeout is used (see Figure 3).

The remote's electronics are based on Maxim Integrated Products's 1-Wire DS2405 and DS2406 switches. Both switches have an open-drain PIO pin that can be controlled remotely. The DS2406 also has integral EEPROM (not used in this design). The DS2405 is used with the valves, and the DS2406 is for the water sense level. I used the DS2406 for water level sensing because its command structure allows for the direct reading of the PIO pin state. The DS2405 can be read only once the command to change it has been



issued. It is an oddity, but it is also much less expensive than the DS2406. So, in a roundabout way, you need to command the DS2405 switch in order to access the PIO state and then possibly command it again to change it to its desired state. I used these for the valve control.

Figure 3 shows dual DS2405 and DS2406 pairs for the remote box and two EarthBoxes. The DS2405 has a 1-kΩ pull-up resistor to its PIO pin, which drives a 2N222A for a red LED on indication for the valve as well as a control input to a MOSFET used to control the solenoid for the valve. The DS2406 is used to read the input from the water sensor. This sensor is simply an open base through 10 kΩ to 1 MΩ to ground to a 2N222A transistor. Both the base and 5 VDC are wired internally to the EarthBox to indicate the proper water level in the EarthBox. When the water touches both leads, the transistor is saturated and the condition is read by the DS2406. In addition, a PNP transistor with a green LED indicates the water condition at the remote box.

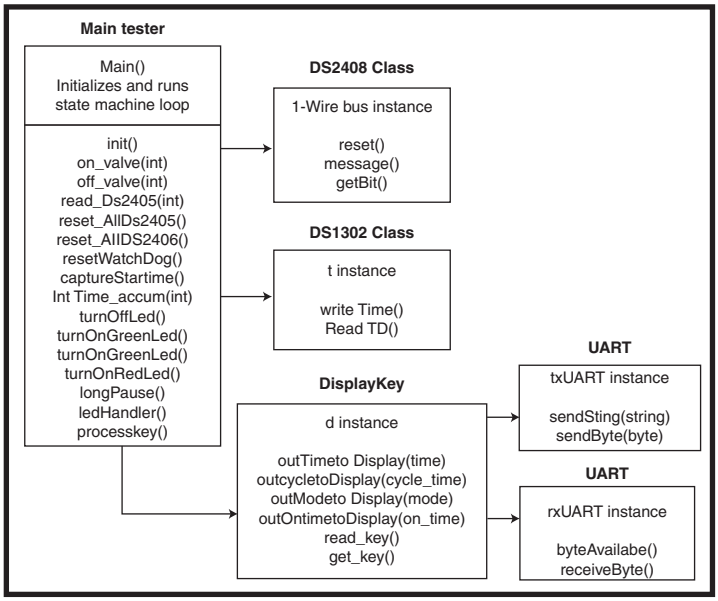


Figure 5—*The controller box's Javelin Stamp used this software structure and these Java classes.*

SOFTWARE OVERVIEW

The system's software is flash programmed into the Javelin Stamp (see Listing 1). There are five distinct classes: `Main Tester`, `Display Key`, `DS2408 Class`, `DS1302 Class`, and `UART`. The first two classes are new for this design. The remaining three are reused from available Javelin Stamp libraries.

The `DS2408` class is used in an instance called `1-WireBus`. The supporting methods (`reset()`, `message()`, and `getBit()`) are all that you need to deal with both the DS2406 and DS2405 once their individual addresses and command sequences are known (see Figure 4).

Each device has a unique factory-set 64-bit address. The addresses can be uncovered in one of two ways. One option is to write a simple Java routine using the `DS2408` class with your DS2408 adapter to pool the address of each part and display it so you can incorporate it in the code. The other is to use Maxim's downloadable iButton-TIMEX software utility. It has a nice GUI for 1-Wire reading and control. It works with Maxim's DS2408 RS-232-to-1-Wire adapter.

The `DS1302` class is used in an instance called `tinstance`. Its associated methods are all you need to set and read the clock. The first custom class, `displayKey`, is simply "d." The function of this class is to build up the display contents, update the mode of operation, time, and status and on times for each valve. This class also reads the associated keypad for operator-requested changes.

Communication to the LCD and keypad is essentially via RS-232. Two instances of `UART` classes are used: one for transmit and the other for receive.

The `Main Tester` class is for the main routine for initialization and system operation. The main tester has methods



for controlling individual DS2405s and DS2406s. It processes recovered keystrokes from `displayKey` and takes appropriate action with display updates, mode changes, and individual valve control. The main tester also manages the watchdog timer and the controller box's red/green LED indicator. In Automatic mode, the main tester functions in a continuous loop (see Figure 5).

LESSONS LEARNED

I quickly made a prototype of the system using the Javelin Stamp and its libraries and capabilities. This technology, as well as 1-Wire, was well suited for the EarthBox application.

The system worked well during the growing season and alleviated the pain of having to water my boxes. In the future, I plan to switch power to the water sensor prior to water measurement. This will minimize any material buildup on the sensor probe due to the 5 VDC. Another alternative is to use an alternating source rather than DC for the water sensor.

Finally, I plan to explore the use of ZigBee in this application to extend the EarthBox's range beyond its 1-Wire limitation. In the meantime, I hope to enjoy my vegetable bounty!

Tom Kibalo (tkibalo@gmail.com) holds an M.S. in Electrical Engineering. He has been a practicing engineer for over 33 years in the Washington, DC area. In addition to engineering, he's an adjunct faculty member at a local community college, where he teaches courses on microcontroller technology. Tom enjoys reading, biking, gardening, and bonsai.

PROJECT FILES

To download code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/201.

RESOURCES

Parallax, "Getting Started with 1-Wire," 2002, www.parallax.com/dl/appnt/jav1/appnote1a.pdf.

———, "Getting started with the DS1302," 2002, www.parallax.com/dl/appnt/jav2/appnote2.pdf.

SOURCES

EarthBox gardening system
EarthBox
www.earthbox.com

KF0604D KYOTTO Solid-state relay
Kytech Electronics
www.kyotto.com

LK204-25 LCD
Matrix Orbital
www. matrixorbital.com

DS1302 TOD chip, DS2405 and DS2406 switches, and DS2408 1-Wire chip
Maxim Integrated Products
www.maxim-ic.com

LM2940 Regulator
National Semiconductor Corp.
www.national.com

Javelin Stamp module
Parallax, Inc.
www.parallax.com

SP232ACP Driver/receiver
Sipex Corp.
www.sipex.com



Build a Three-Axis CNC Mill Machine

Gordon shows you how to design and build a computer-controlled wood mill machine. The system includes an old medical X-ray machine, a Galil DMC-2133 three-axis intelligent motion controller, and an old 386 laptop.

I built my first computer-controlled XYZ router machine about 12 years ago ("Designing an Industrial Grade XYZ Router," *Circuit Cellar* 62, 1995). My intention at the time was to create signage using wood or acrylic. As a result, the large machine had a 50″ × 50″ workspace. For various reasons, my attempt to get into the sign business failed, and I was left with a huge machine that took up nearly half of my once-spacious shop. It was certainly a valuable tool, but it was not making me rich. So, I decided that I needed to sell it to make room for a much smaller machine.

At about the same time, during one of my frequent visits to a government surplus outlet, I found a medical X-ray machine that had two motion stages that could be modified for use in my second appliance. One stage even had a lead screw in it. I don't remember the exact price of the unit at auction, but it was less than $50. And I knew that it would save so much work when it came time to build my newer, smaller machine. So, it was soon stripped down to only the parts that I could use and put in storage. It sat in storage for several years while I pursued a variety of other projects and before I got serious about building the second machine. Now, about a year after starting, I have a finished system.

DESIGN CONCEPT

When building for myself, my designs are influenced strongly by the materials and parts that I have on hand. For this project, I had servo motors (4″ D × 7″ L with a 0.625″ shaft) from some mainframe tape drives. I also had linear servo amplifiers for the motors (from the same

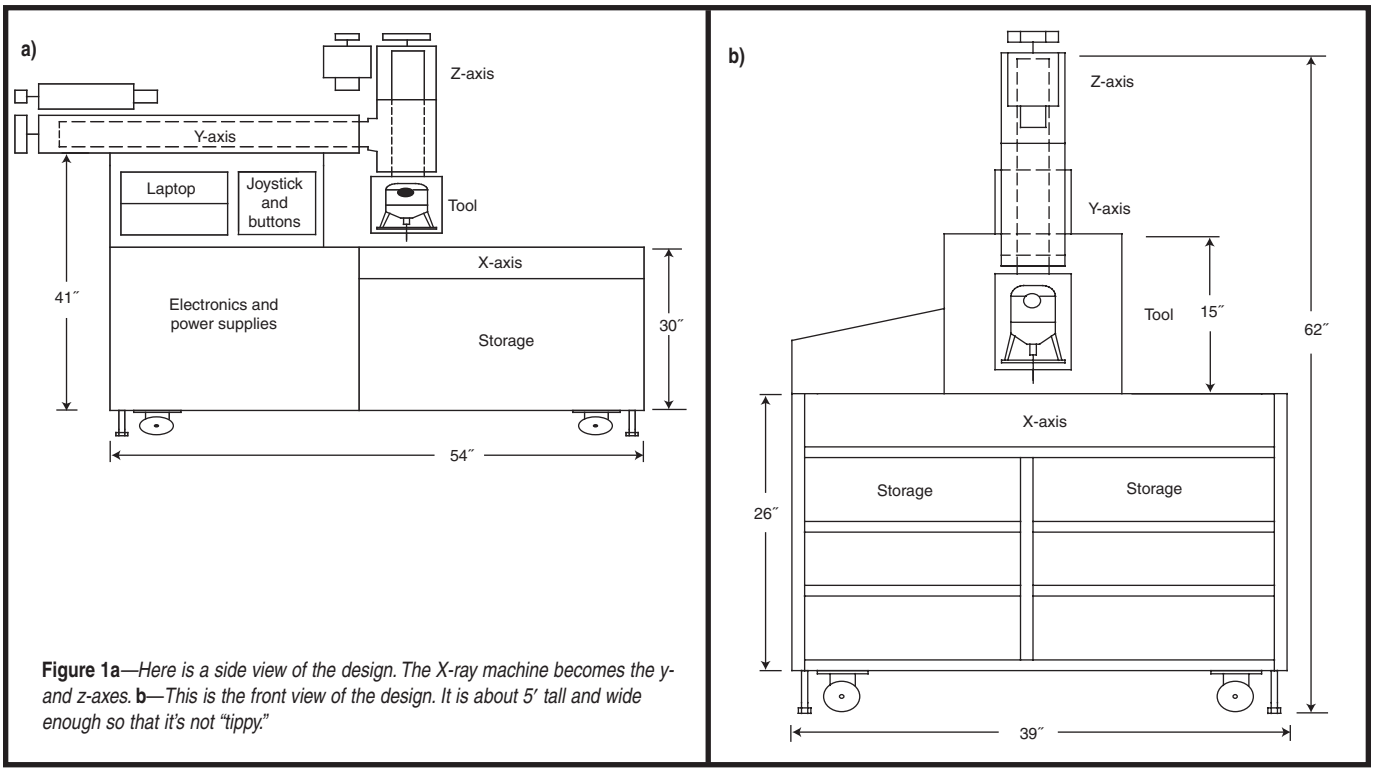


Figure 1a—*Here is a side view of the design. The X-ray machine becomes the y- and z-axes.* **b**—*This is the front view of the design. It is about 5′ tall and wide enough so that it's not "tippy."*

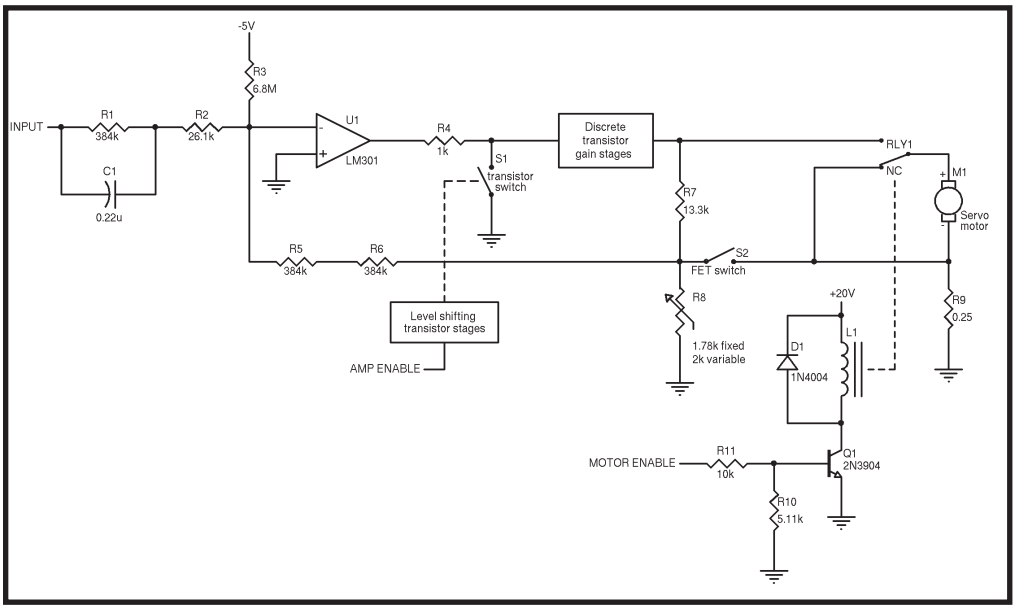


Figure 2—*This is a simplified schematic of the tape drive servo amplifier. Without the original schematic and manuals, I probably wouldn't have used these amplifiers.*

that evolved after a few iterations is shown in Figure 1. Some dimensions were included to allow for a correct overall size impression.

I initially planned to make the frame of the cabinet out of thin wall-square tubing and then cover the frame with sheet metal. A few things stopped me from doing this. I didn't have a MIG welder to build a metal cabinet, and I didn't want to contract the job out. In addition, I believed the metal version would be more difficult and expensive to change. So, the cabinet was built out of 0.75″ birch plywood and designed to be as mechanically rigid as it is practical.

tape drives), a power supply with an aluminum enclosure for the parts, two motion stages from a medical X-ray machine, several 386 laptops, sprockets and a chain from another motion-control machine, and casters from a printer or a mainframe.

In addition to incorporating the aforementioned parts, I had other design criteria. The machine had to occupy as small a footprint in my shop as possible and it had to be modular enough that it could be moved to another shop in the event I moved or sold it. In other words, the modules had to be able to pass through a typical door. The workspace had to be about 20″ × 20″. Incremental optical encoders had to be attached to the motors. A joystick had to be part of the control panel. The machine had to be on casters and also have screw feet. And lastly, the machine had to use a Galil Motion Control DMC-2133 intelligent motion controller. (I had used these before and was familiar with them. More on this later.)

With these criteria in mind, I set about designing my computer-controlled wood mill. First, I needed to take some measurements of the X-ray machine's parts. I found at that point that the motion stage travel distance of what would eventually be the y-axis was 18″. This was a little shorter than I wanted, but since I was both the customer and the vendor, I could change the specifications. The travel on what would become the z-axis is 9″, which is just fine for this machine. Since the X-ray machine provided only two axes, the x-axis had to be designed and built from scratch.

It was also during this time that I discovered that the motion stage for the y-axis would need more modifications than I first thought. Some crucial bearings and bearing tracks that weren't necessary on the X-ray machine had to be added. Without going into detail here, let me just say that this added considerable time and effort to make the motion stage as useful as the y-axis. However, the work on this part proceeded simultaneously with the overall machine design. So, when I got tired of working with the CAD package on the conceptual and detailed design, I went out to the shop and worked on the motion-stage modifications. The conceptual design

DESIGN & CONSTRUCTION

With the conceptual design finalized, I began the detailed design. Several areas needed to be addressed and it's probably not appropriate for me to show you drawings of all the parts. Rather, let me list the areas that required detailed design. The z-axis motion stage was not driven before, so a lead screw, a bearing support structure for the lead screw and chain drive, and a motor-mounting plate were required. A means of attaching a tool to the z-axis was also needed. The y-axis motion stage needed a different lead screw and a motor-mounting



Photo 1—*Operation of the machine would normally be done from this side. The podium is sized to put the laptop screen close to eye level.*



Photo 2—*Here's a view from the opposite side. The controlled AC outlet and the arm to support the moving cable are shown.*

plate. I had to build an x-axis motion stage and the motors needed encoders mounted on them.

After many hours (days, weeks) in front of the computer building parts on virtual paper, construction could finally begin. And construction proceeded with no major difficulties to completion. One minor issue did come up however. When changing the y-axis lead screw, I discovered a large amount of backlash in the existing nut assembly. The amount wasn't measured, but I'm guessing now that it was close to 0.125". With a minor redesign of the nut-restraining mechanism, the backlash was reduced to an acceptable level. (Actual backlash measurements will be given later.)

Since the backlash has been mentioned here, I should also point out that toothed drive belts would have less backlash than the chains I used. Had I built this machine with all new parts, toothed belts would've been my choice. I knew that the intelligent motion-control card was going to cost me over $2,300, so I was trying to keep other costs down wherever possible. This machine has no travel limit switches or home switches. Those features were included on my first machine, but I found them to be of little use. However, some provisions have been made so they could be installed later on this machine.

MOTION CONTROLLERS

The DMC-2133 motion controller has nonvolatile memory that enables it to store motion programs that can be executed on power-up. So, no desktop computer is required, other than for initial programming. Of course, if the machine is to be used in applica-



Photo 3—*When viewed from the front, you can see the ample storage area. The x-axis is one of its travel limits.*



tions where the motion program changes frequently, such as the wood mill described here, it makes sense for a computer to be permanently attached to the controller. Since the function of the computer used in conjunction with an intelligent motion controller is usually only for communication (i.e., transferring and executing motion programs), its capabilities need only be minimal. I used an old 386 laptop that ran DOS, something most folks would have thrown out long ago.

In addition to providing the "brains" to perform the calculations necessary to implement a digital control loop, the intelligent motion controllers provide digital I/O. Digital outputs are useful to control a cutting tool or a dust collector. Digital inputs are helpful for providing signals to the controller (e.g., starting a motion program on a button press). Intelligent motion controllers can also be supplied with ADCs. This makes controlling a machine with a joystick very easy to implement. Galil's family of controllers is programmed using a language specific to them, very similar to assembler. This language is extensive in terms of how the behavior of the controller can be changed. The one I used has a separate, rather thick command summary manual, containing the syntax for 192 commands. Fortunately, you don't have to learn all the commands, but a dozen or so need to be learned, and you need to be aware of what is possible with the others. Simple tool paths can be programmed manually, but complicated parts require a different approach. (More on this later.)



Photo 5—*This is the back of the machine. Digital panel meters monitor the system DC voltages.*

MODIFICATIONS

With construction complete, the last phase could begin: the wiring. One area that always requires special



Photo 4—*I removed the worktable! The x-axis drive motor is partly visible in the rectangular cutout.*



consideration is the cables in motion. They must be allowed to move, but they must move in such a way that the copper doesn't fatigue and fail. And, of course, the cables must be protected from physical damage. The advantage of this design is that only the z-axis cables are in motion.

Commercial machines use a variety of clever techniques. Most require special hardware to guide and protect the cables. The method I chose to use is effective and didn't require buying any hardware. A spring-loaded arm mounted on a pivot plays out or takes up cable as required.

I doubt I would've used the servo amplifiers from the magnetic tape drive if I didn't also have the schematics. Reverse engineering a circuit board to obtain a schematic is a daunting task, and without a schematic there's no way to know how to connect the amplifier. Because I had schematics, I was able to make several modifications to the servo amplifiers to better suit them to this application.

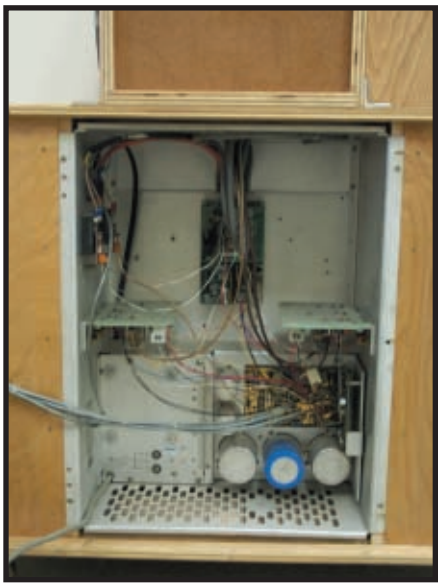
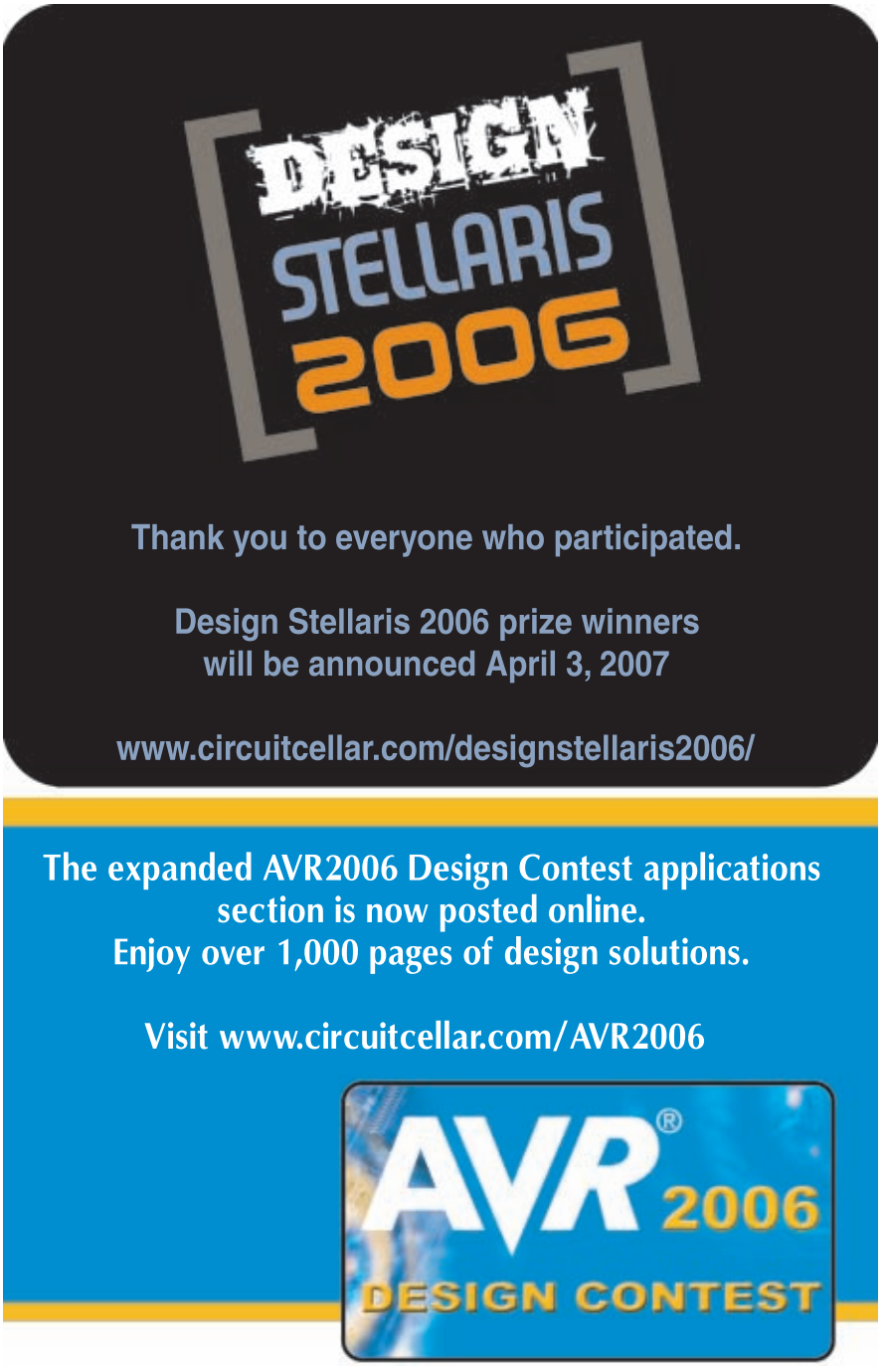


Photo 6—*With the front panel removed, the inside of the electronics cabinet can be examined. Who would guess that this aluminum box used to be the home of a magnetic tape drive?*

The original servo-amplifier schematic is considerably "busier" than what you see in Figure 2; however, this simplified version is better for explanation. The original photocell circuit that produced input to the servo amplifiers from dancer-arm motion must have introduced some offset, because both the supply and the take-up reel amplifiers had an offset introduced via R3. I removed R3 because I didn't want the offset here.

Switch S2 is a JFET, which is controlled by some circuitry external to the servo amp. When S2 is open, U1 has voltage-shunt feedback applied to it via R5/R6 with the voltage on the right-hand side of R6 determined by the R7/R8 voltage divider. This makes the servo amp a transvoltage amplifier, and as the 2-kΩ part of R8 varies through its range, the grain varies from –9 to –17. When S2 is closed, the voltage on the right-hand side of R6 is determined by the voltage across the current-sampling resistor, R9. Although the R7/R8 divider is still present, the voltage dominates it across R9. Since the voltage driving the R5/R6 resistor is derived from a current, U1 now has current-shunt feedback applied to it. The servo amp is best described as a transconductance amplifier with a gain of –8 A/V. I hard-



wired a jumper across S2 to make the amplifier permanently transconductant.

A relay, such as RLY1, is often used between a servo amp and the motor it drives. When the relay is deenergized, the NC contacts place a short across the motor terminals, providing very effective braking. When the motor is to be driven, the relay is energized and the motor is connected to the amplifier. I like this kind of control to be activated by limit switches or, in my case, manually by a switch on the control panel.

The original servo-amp PCB had some components that interconnected the AMP ENABLE and MOTOR ENABLE circuits. I removed some components and added R11 to separate their functions. The AMP ENABLE input is driven by a signal from the Galil board. When the circuitry associated with S1 is driven, a short is applied to the output of U1 through a 1-kΩ limiting resistor. This disables the power stage.

COMMISSIONING

Powering up a closed-loop motion control system for the first time must be done carefully. There are many places where signal inversions can be introduced when the wiring is being done, so there's a good possibility that the system will initially run away. If the motors used are large and can produce significant motion-stage velocity, machine damage is a distinct possibility. Intelligent motion controllers have features that allow the initial power-up to be done without any scary machine runaway.



Photo 7—*Check out what this machine can do. The metal ruler on the bottom is for a size reference (12″). Finding vector-based fonts for letters and numbers was a problem in the past.*

The motion controller can be programmed to remove the drive to the servo amps if a significant position error develops, which is extremely helpful when powering up a system for the first time. In addition, if the sense of the feedback loop is incorrect, there are commands to invert the sense of the feedback from the encoders and invert the behavior of the motor. It is likely that no wiring changes will be necessary to correct a runaway situation.

Even when the sense of the feedback is correct, the control system may be dynamically unstable. This was the case with my machine. When a feedback system involving mechanical components oscillates, it often manifests as a high-pitched "whine." Other behaviors are possible, including the classic large-amplitude sinusoidal oscillation. So, when I heard the whine coming from my motors, I wasn't happy. The connection of an oscilloscope confirmed there were large amplitude oscillations in the motor-drive signal, which got worse at the





Feature Creep Welcome

Successful Designs *and* Redesigns Begin with Rabbit

RCM4000 Family

$72 (qty. 100)



- 10Base-T Ethernet
- 32 MB NAND flash
- Up to 25 digital I/O

RCM4100 Family

$61 (qty. 100)



- 12-bit A/D model
- Industrial temp. spec.
- Up to 40 digital I/O

RCM4200 Family

$81 (qty. 100)



- 10/100 Fast-Ethernet
- 8 MB Serial flash
- Up to 35 digital I/O



Development Kits

From $199

Easy Integration with Design Peace-of-mind

The powerful Rabbit 4000 based core modules are the perfect choice for your next embedded design. That's because with each family being pin-compatible with the next, you can design your product once and change features as simply as changing modules. So if you need Ethernet, Fast-Ethernet, analog, removable memory, or wireless connectivity, we have an easy solution that's perfect for you.

Order Your Kit Now

Development kits include everything to start your development, powerful Dynamic C development software, RabbitCore module, development board, programming cables and accessories.

www.rabbit4000.com

RABBIT® Semiconductor

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

start and end of a move.

Initial machine power-up is usually done with the controller set for a low value of K_p and no K_i or K_d. (K_p is proportional gain. K_i is the integral gain. K_d is the derivative gain.) Reducing K_p didn't have much of an effect on the oscillations, which was strange. After some head scratching, I went back to the servo-amplifier schematic and found the problem I should've seen before. Resistor R1 has a capacitor across it (see Figure 2). This drastically increases the high-frequency gain of the servo amp, not usually desirable from a stability point of view. When C1 was removed, the system was fairly well behaved. At this point, I also hardwired a jumper across R5 to reduce servo-amp gain to –4 A/V, which further improved system behavior.

With the changes to the servo amplifier feedback network complete on all three axes, the system was well behaved. I finally had a working machine!

FINISHED MACHINE

The laptop sits on a podium to bring the screen to eye level and to slant the keyboard (see Photo 1). A small control panel containing the joystick and switches also sits on the podium. To make the joystick operate the machine, I had to write a motion-control program. This is an example of why it is necessary to be familiar with some of the intelligent controller commands. When the joystick program runs, moving the joystick from side to side controls the y-axis and moving it back and forth controls the x-axis. Short movements of the joystick move the corresponding stage slowly, whereas long joystick movements increase the speed of the stage. The joystick is an expensive, well-built military surplus part.

Being able to move the machine about the workspace using a joystick is a significant feature. Previously, getting the machine into the proper starting position was tedious. All machine motion was initiated from the keyboard. And on the predecessor, the keyboard was several steps from the machine. Just getting the tool into the starting position was time consuming.

	x-axis	y-axis	z-axis
Repeatability error	0	0	0
Accuracy	0.003″	0.001″	–0.003″
Backlash	0.005″	0.018″	0.008″

Table 1—*The machine's performance is respectable. The y-axis backlash could be improved, but that would require significant disassembly just to get at the source. Unless it becomes a problem, it will remain as is.*

Since the joystick produces only two analog signals, it can't control the z-axis using the same method. It may not be obvious in the photo, but the joystick has a push button on the top of it. Pressing the button makes the z-axis move at a fixed velocity in a direction established by a nearby toggle switch.

The red button in the upper-left corner of the control panel is the master Reset for the controller. It could also function as an Emergency Stop button. A toggle switch on the control panel is the actual Emergency Stop. It controls a set of relays that are part of the servo amps that can disconnect the motors. Also, there is a momentary toggle switch that is used as a Start signal. I try not to let any motion program execute directly from the keyboard. Having to push a Start button allows for one last check to make sure all is well before a motion program begins. The arm to deal with the moving cable pivots slightly to allow the stage to move out.

Wooden covers enclose the chains and sprockets on all three stages. In addition to being a safety feature, the covers also keep out dust. They protect the sprockets during transportation as well (see Photo 2).

A riser box elevates the y- and z-axes. There are several stiffeners inside the box to make it resistant to racking and twisting. The box is not permanently attached to the lower part of the cabinet. When the system was almost complete, I decided the y- and z-axes weren't quite high enough, so I added an additional 2.25″ riser by sandwiching three layers of 0.75″ plywood. (I did this to avoid making a new riser box.) Note that the z-axis cables have been retracted by the pivoting arm. The gray aluminum panel under the outlet is where the main AC power for the machine is connected.

The intelligent motion controller controls (through a solid-state relay) the AC outlet in the middle of the cabinet. I have used a similar approach to control a shop vacuum for dust collection. The dust-collection approach on this machine is still under development.

To prevent the system from overheating, a rectangular hole was installed to the right of the outlet with the aluminum mesh to cool the electronics cabinet behind it. There is a similar ventilation hole on the operator side. There are also two holes in the bottom of the cabinet with screens to allow air convection.

A blank for a part is attached to the worktable using a collection of shop-made clamps (and blocks sometimes) fixed via 0.25″ cap screws and captive nuts on the back. The worktable can be replaced should it get eroded by the tool, but with care, that can be avoided. The area under the x-axis is storage space for clamps and manuals. I might add doors on the front to keep out dust. Note that the top shelf under the right-hand side of the x-axis is smaller than the others because the drive motor hangs down under the stage (see Photo 3).

The design of the motion stage is conventional with one minor exception. Usually, you would find hardened steel rails with recirculating ball bearings. But those are rather expensive, and I didn't think they were necessary for this machine, because the accuracy requirements for wood parts are more relaxed. So, the rails are ordinary cold-rolled steel and the bearings are bronze bushings with oiling wicks. The lead screw is a 0.625″ acme-type thread rod sold in 3′ lengths and selected for straightness. (It isn't actually intended to be used as a lead screw.)

The x-axis motion stage is surprisingly easy to move despite the lack of recirculating ball bearings. Initial indications are that the bushings seem to be quite tolerant of small amounts of dust, although my intent is to keep as much dust off them as possible. (More on that later.) Elevating the worktable so it clears the sides of the cabinet when nearing the travel limits is accomplished with the wood risers attached to the motion stage. The

wood risers touch the bronze bushings so there is some oil staining, which might be visible (see Photo 4).

Just below the motion stage, near the front, there is a slot cut in the cabinet. This slot will accommodate the installation of the x-axis limit and home switches. A finger will extend from the motion stage through the slot and the switches will mount on the underside of the slot.

The machine's cabinet is designed as two closed boxes. One is the riser block; the other is the lower cabinet. However, to install the electronics cabinet, there had to be a large hole in the back of the lower box. In order to retain as much strength as possible, the electronics cabinet's cover panel also doubles as one side of the lower box. To that end, there are five screws (1.5″ long) on each side of the aluminum panel that extend through the plywood into a hardwood backing plate to try to retain the integrity of the face of the lower box (see Photo 5).

Digital panel meters monitor the DC voltages produced by the power supply from the magnetic tape drive. The machine is powered up from here. The power supply produces 5 V for powering the intelligent motion controller, ±12 V for the joystick and the intelligent motion controller, and ±20 V for the servo amplifiers. This power supply can also deliver ±40 V to the servo amps under logic control, but at the moment I haven't made use of that feature. Although I'm not expecting a power supply failure, it's comforting to see the correct DC voltages displayed when the machine is powered up. The main machine's On/Off switch is in the lower-right corner of the aluminum front panel.

There is still a lot of empty space in the electronics cabinet. Had there been a shortage of space, I would've repackaged the power supply and servo amps. But in order to get the worktable at a suitable height, the cabinet had to be large, which meant the electronics cabinet from the magnetic tape drive could be used "as is." This saved me a significant amount of work (see Photo 6). The intelligent motion controller is mounted on the back of the cabinet and cables bring encoder signals to it.

There is a servo amplifier assembly mounted on each side of the cabinet, with the side of the cabinet becoming part of the servo-amp heatsink. There are actually two servo amps in each assembly, so one amplifier is unused. The solid-state relay for the controlled outlet is on the left side of the cabinet above the servo amp. There is a second solid-state relay behind it for future use. The very top left-hand corner is where an AC power distribution terminal block is located. The lower third of the cabinet is occupied by the power supply. Since the power supply provides the logic power for a rather expensive intelligent motion control card, I am pleased that it has output over-voltage protection. That's a fairly common, but necessary, feature here.

I'm not particularly proud of the wiring inside of the electronics cabinet, some of it would be called "point-to-point," as opposed to being nicely bundled into cables and clamped. At the time I thought no one would ever see it, so why fuss with it? I guess I was wrong. The cable on the bottom left in Photo 6 goes to the On/Off switch for cabinet power. The bundle of cables above it goes to the digital panel meters.

I produced a collection of items on the first machine. They include the number "2" done in cedar (about 0.625″ thick), a gray letter "M," a tan cutting template from countertop plastic laminate, the letter "A" in smoked Plexiglas (0.125″ thick), a business card holder with a 0.3125″ deep pocket, a plaque showing a curling rock, and a floppy disk holder with 0.25″ deep slots (see Photo 7). The floppy disk holder is done in walnut, which is dark, so the slots are not clearly visible. The second machine is still so new that I haven't made anything of note yet.

The optimum depth of cut for the router was 0.125″. Any time more depth was required it was done in multiple passes. This machine has superb repeatability, so that wasn't a problem. The feed rate I use most of the time is 0.5″ per second.

In the letter "A," a piece is inserted where there is normally a cutout. This was done as a demonstration of the machine's capability. When I was making these letters, I ruined a few before I realized I had to leave "tabs" connecting the letter to the blank. Otherwise, when the letter came free at the end of the cutting, it would shift and be caught by the bit, always causing fatal damage to the letter. Since I was using them as samples for a mail out, they had to be flawless.

MOTION PROGRAM DEVELOPMENT

Galil has produced a utility package that allows motion control programs to be produced in a highly automated fashion. It's called CADTODMC. This utility allows the tool path from a drawing package to be converted to Galil motion control commands with a desktop computer. The drawing package needs to have the ability to produce a DXF version of the tool path.

The motion-program-development steps are: create the drawing of the tool path using a suitable CAD package, convert the tool path drawing to a DXF file, and use CADTODMC on the DXF file to create the motion program. This process appears simple and eventually it is. But, initially, there is a lot to learn, and while learning is happening, mistakes are being made that can be frustrating to correct.

PERFORMANCE MEASUREMENTS

How good is this machine? The measurements I'll describe next are the first I've done on a machine that I've built. Earlier, there was no reason for measurements because the machines adequately did what I wanted. Most parts didn't require absolute accuracy; however, good repeatability was often important when making multiples of the same part. For example, I have stacked up multiples many times and I have been pleased at the excellent repeatability. Before presenting a summary of the performance specs, let me explain how the data was obtained. All measurements were made using a dial indicator capable of measuring just over 1″ of total travel, having a scale calibrated in 0.001″ and which can be interpolated sensibly to distinguish units of 0.0005″.

You should also appreciate that per-

formance is largely dictated by the mechanical system and not the digital control system. This machine uses optical encoders with 128 pulses per revolution. The Galil board introduces a revolution improvement by a factor of four. When the lead-screw pitch and chain drive ratio are taken into account, this translates into 5,120 quadrature counts (QC) per inch or 195 µinches/QC of basic system resolution. Unless some K_i is used, there will be some steady-state positioning that's error-dependent on forward gain. In this case ±2 QC, or an error of 390 µinches, which is slightly less than my measurement capability of 0.0005".

To measure repeatability, moves to the same location were repeated and the dial indicator reading was noted. Different dial indictor readings indicated a lack of repeatability. It came as no surprise to me when I wasn't able to measure any deviation from perfect repeatability. To measure accuracy, a move of exactly 1" was executed from a zero (on the dial indicator) position. This move was done several times with the measured error being the same on the respective axis. I expected these errors to be larger than they were, just because the lead screws weren't actual "lead screw" quality. Some errors in cutting the lead screw threads would easily account for this positioning error.

My first attempt at measuring the backlash gave poor results. My best results were obtained when I physically pushed a stage back and forth and noted the dial indicator reading. Since the method of trapping the lead screw nut is different on the y-axis, I expected the backlash to be larger there. But on all three axes, the backlash was lower than I expected. Overall, I'm very pleased with the performance of my machine (see Table 1).

IMPROVEMENTS

It seems to be a human trait to look for ways to improve machines. My second machine is better than the first in many ways. Since you don't have intimate knowledge of the first machine, I won't talk about those improvements. But, I will talk about improvements and enhancements that would be appropriate for this machine. I have years of experience building things out of metal and wood. So, generally my intuition about what will be strong is pretty good. However, when I chose 0.75" cold-rolled metal for the rails of the x-axis, that wasn't strong enough. It probably isn't important in this application, but the x-axis stage has more "give" for vertical loads than I would like. Certainly, I would use heavier rails in another version.

At the moment there is no means of dust collection. That's rather important for a machine like this because the dust produced is very fine, and there is lots of it. Previously, I used a shop vacuum with its hose attached to a shroud around the base of the router. This machine doesn't use the base of the router, so something different needs to be done. My first attempt at collecting dust on this machine was unsuccessful. There is more work to do in this area.

My original design called for all of the critical parts of the motion stages to be completely enclosed with accordion-like jackets. Bearings rolling on even small amounts of dust soon accumulate lumps of packed dust. Making these jackets is somewhat difficult, so I opted for "dust skirts" on the x-axis only. These are attached to the sides of the worktable with some weight at the ends. (The photos were all taken before the dust skirts were installed.) So any dust not caught by the dust-collection scheme will fall on the skirt rather than on the x-axis rails. The y-axis and (the z-axis to a lesser extent) will still be subject to airborne dust, which I know will be a minor problem.

I have not spent much time "tuning" the controller. Some adjustment of the controller parameters had to be done when the system was first powered up just to get the system stable from a dynamic point of view. I haven't gone beyond that, but I should. Galil has some slick software to help with the tuning process, which I will likely use.

The last improvement I'll talk about is the most interesting. System backlash can be compensated for by using what's called "dual-loop control." This means there are two position feedback loops per axis and therefore two position sensors per axis. One is usually on the motor shaft and it is usually an optical encoder. The other is on the motion stage and it is usually a translational transducer. The controller then uses the rotary optical encoder for "rough" positioning and the translation transducer for "final" positioning. Since the final position is determined by the actual position of the motion stage, the error due to backlash is eliminated.

These translational transducers tend to be expensive. But I was recently given three resolvers taken from an old NC mill that was being scrapped. With a little work and some cables and pulleys, they can be made into translational units. Here's the best part: the DMC-2133 was designed to implement dual-loop control. I plan to modify this machine for dual-loop control, but I have not set a target date. There is another aspect to using resolvers with intelligent motion controllers. In general, they don't accept resolver signals as inputs. They are designed for optical encoders. The resolver signals must be converted, so they are similar to optical encoder signals in order to use them.

Gordon Dick (P. Eng.) is an instructor in Computer Engineering Technology at the Northern Alberta Institute of Technology in Edmonton. Occasionally, he consults in intelligent motion control. Gordon is usually working on microcontroller-based projects (often involving machining of metal parts). You may contact him at gordond@nait.ab.ca.

PROJECT FILES

To download code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/201.

RESOURCE

Galil Motion Control, Inc., "CAD-to-DMC User Manual," www.galilmc.com/support/manuals/cadman.pdf.

SOURCE

DMC-2133 Motion controller
Galil Motion Control, Inc.
www.galilmc.com

Reverse-Engineered ECP Bus

Miguel wanted to build a system to simulate strokes on a keypad so he could trigger modes on an alarm system. But first he had to learn about ECP buses. In this article, he explains what he learned.

When trying to gather information about a new chip, protocol, or device, a search engine can often be your friend. Just knowing the name of a given system may be enough to find all of the information you need. When you are lucky, searching on the Internet can lead you to a variety of documents and useful information. However, this is not always the case. It was definitely not true this time. Sometimes you need to interface with systems that are poorly documented or not documented at all. Whether the lack of documentation is anybody's fault or it is the true will of the manufacturer, you may be in for a big dose of guesswork.

In my project, the goal was to simulate keystrokes on a keypad so I could trigger the desired operational mode on an alarm system. The problem was that I knew nothing about the proprietary bus that the system used, and I had only a sample working system to play with. I decided to look for a PC-based data capture system. With the system, I obtained samples of the communication. Next, I created a way to display the long list of samples I received by combining a small piece of Java code with a fantastic piece of vector-graphics software. I also learned a bit about scalable vector graphic (SVG) language during the process.

Having samples of the communication you want to emulate can help you figure out how it works. I continued to write more Java code to decode the captured data once I knew what the format was. Finally, I wrote some Perl code to allow a PC to emulate the communication that I was asked to perform. Later on, I turned that into a small C program for a Rabbit core. You will find all of these programs and some of the sampled data on the *Circuit Cellar* FTP site. Now I will show you how I did it.

ECP BUS

I am sure that most of you have heard of an extended capabilities port (ECP) interface. The ECP was designed by Hewlett-Packard and Microsoft to provide a new high-speed version of the standard parallel port (also known as Centronics) found in early PCs.

The bus in my project was a serial bus, and it had nothing to do with personal computers. The best information I found on the Internet was that the ECP bus consists of four wire interconnects: power, ground, data transmit, and data receive. The information, while true, did not help much. Because I was not finding useful information on the Internet, I had to resort to other sources. So, I talked to some of the people who install these security systems.

A security company representative told me that ECP buses serve to interface several security panels, consisting of a keypad and a display, to an alarm motherboard. An ECP bus is a multidrop network that also provides the power supply for the panels. An alarm motherboard can be operated from several remote panels. For example, a house may have one panel in the garage and another one near the main door. If you watch someone typing a code in a home security system in a movie, chances are it is one of the panels that I am talking about.



Photo 1—*USB Instruments's Stingray consists of two input channels and one signal generator.*

Some people on USENET news also asked about details on the ECP bus protocol, but no answers were provided. I contacted two people that showed interest in the topic, but I obtained no information. I even found U.S. patent #6868493 that referenced the ECP bus, but it did not give specific details. While I am sure there has to be a way to pull

the bus specification from the manufacturer, I did not envision a fast, straightforward method at the beginning of the project. I just had to figure it out from scratch. I started by looking at signals I could pick up from the bus while operating my sample system. Later on, I figured out how the system works by looking at the captured information.

GATHERING DATA

I do not perform reverse engineering for a living, so I was not well equipped to carry out this task. I was dealing with 30-year-old equipment and I assumed data transmission speeds would not be very high. I needed to obtain a good sample of the signals over the transmit and receive wires of the ECP bus so I could start figuring things out.

A sample oscilloscope would be all right, but only for very basic tasks. It was also not clear if a PC serial port could handle serial communication over the bus. Using an old oscilloscope, I could observe that bus signals ranged from approximately 0.8 to 11 V, while the power signal was 12 VDC. Using my analog oscilloscope, I was not getting a good idea of the bus's data rate or format, so I needed a new gadget to get the signals properly sampled and stored for further analysis. I thought a PC-based signal-capture board would be appropriate, but after a quick search on the Internet, I learned that the USB Instruments DS1M12 Stingray dual 12-bit oscilloscope was available from a local seller at a good price, so I bought it (see Photo 1).

Besides the Windows software provided with the unit, there is a DLL you can use to gather data from your own software. (By the time I learned about this it was too late because I had created a workaround first.) I was glad to learn that the unit could also be used under the Linux OS and that the Windows-based software would run on top of Wine too.

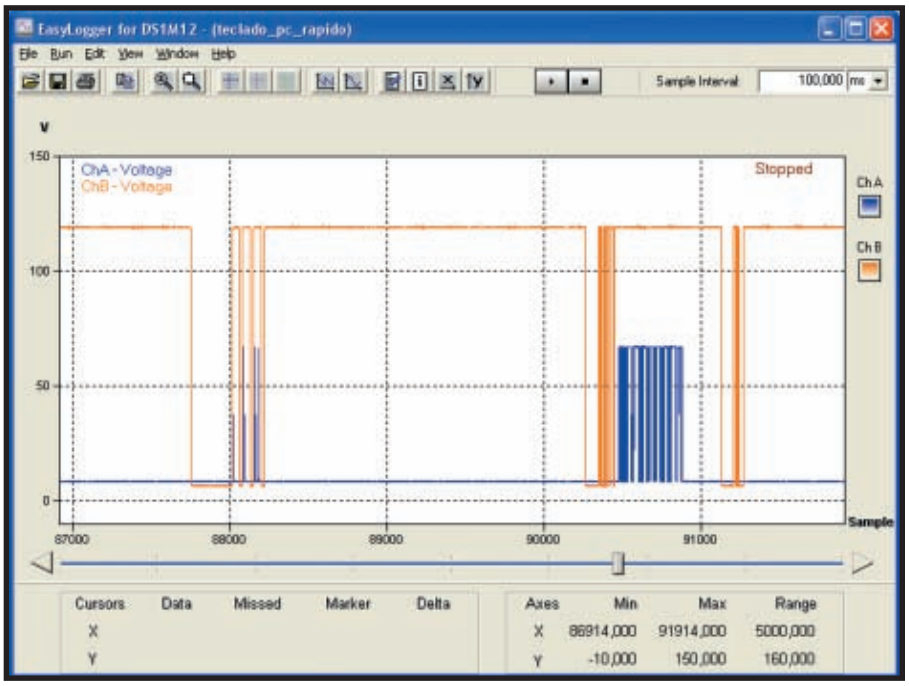


Photo 2—*EasyLog software comes with the Stingray. I used part of a captured signal here.*

The Stingray came with two programs: a sampling oscilloscope and a data logger. The latter was designed to capture long sampling sessions (see Photo 2). It allowed me to view and save gathered data to a file. It would also allow me to save the sampled data to a text file. It seemed quite appropriate to pass the gathered data on to another application for further processing.

Once transmit and receive signals had been gathered while operating the keyboard, I turned to the process of finding out how it works. I had my text file with a table-like format with measured voltage for each signal sampled at 20-µs intervals. Just looking at the captured signal, it became obvious that both transmit and receive were baseband digital signals using non-return-to-zero (NRZ) encoding. However, it was not clear whether asynchronous serial communication was used or not. Signal polarity might be inverted too.

I first needed to know the signal's data rate. For an NRZ-encoded bit stream, you need to look for the shortest length of a signal element to discover the bit time. In my case, the bit time was around 200 µs. I knew that because it was a value between 10 and 11 samples. However, most of the time, serial communication uses certain data rates that can be easily generated from a crystal oscillator. These are values like 300, 1,200, 2,400, 4,800, 9,600, 19,200, or 38,400 bps. Assuming the bus's data rate was one of them, I just needed to calculate the inverse of these values to know the bit time for each one. It turns out that 1/4,800 is 208 µs, so it would be just this value. Once I guessed the bus's bit rate, the next step was to find the data format. Is asynchronous serial transmission being used? If yes, what is the number of data and stop bits? What parity, if any, is being used?

Asynchronous serial communication keeps the line idling to a high level while no data is available and it inserts a zero bit to begin transmitting

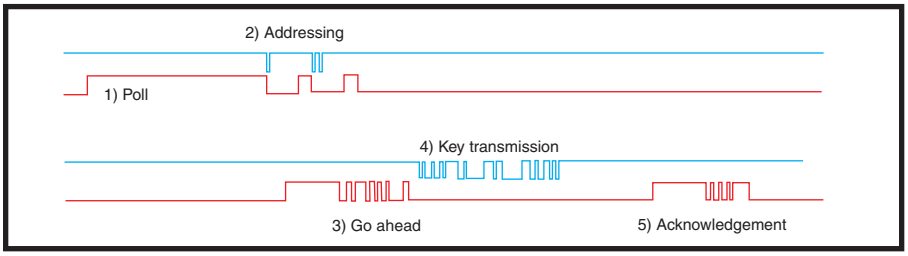


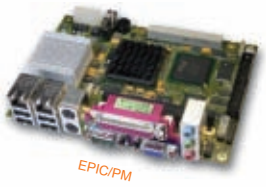
Figure 1—*Check out the details on the protocol associated with the transmission of a key press.*

TRI-M SYSTEMS

Proudly distributes:

kontron

Intel® Pentium® M 745 1.8GHz, 2MBL2, ATX



EPIC/PM

Based on Intel® Pentium/Celeron® M processor.

Intel 855GME / ICH4 chipset with integrated Intel® Extreme Graphics 2 engine 2x32 MB VRAM UMA.

2x DDR-RAM-SODIMM for up to 2GByte.

TRI-M ENGINEERING

IDE Flash Drive Carrier Board with Micro SD Interface



SD-IDE-40/44

Available in 40 and 44 pin header configuration

Support PIO 0-4 and Ultra DMA 3 mode.

Bootable from Transflash/micro SD.

Low power consumption.

Evalue Technology Inc.

Fanless Intel® 852GM Celeron® M 600 MHz Micro PC



EES-3723

Onboard Intel® Celeron® M 600 MHz with 512K L2 Cache CPU.

Intel® 82852GM Chipset.

Scalable with Evalue EPIC SBC.

One SODIMM up to 1 GB DDR SDRAM.

www.tri-m.com info@tri-m.com

1.800.665.5600

HEAD OFFICE: VANCOUVER

tel: 604.945.9565 fax: 604.945.9566

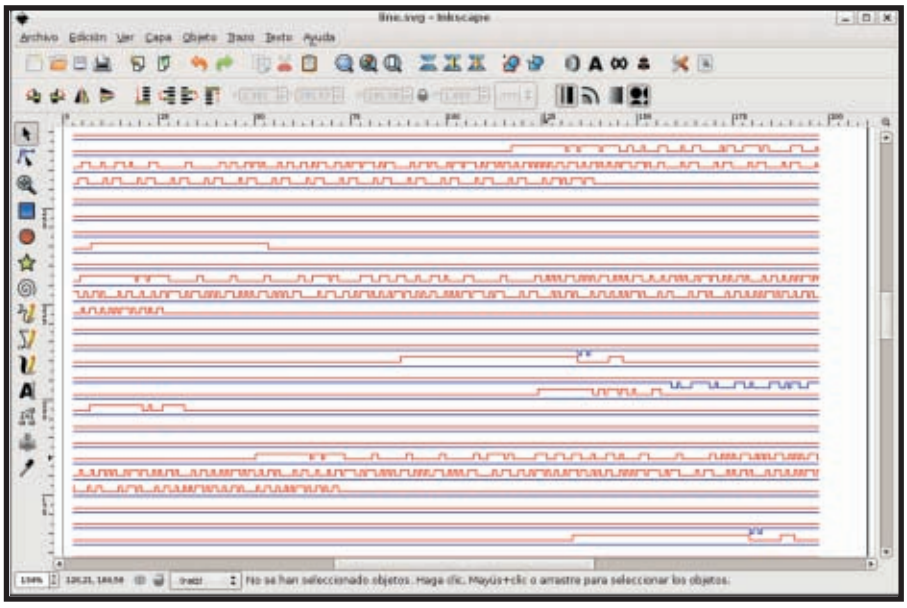


Photo 3—*The Inkscape program shows part of the drawing generated by the Draw.java program.*

a character. However, by scrolling the captured signals back and forth using the data logger software, it was not clear whether they were even the same in terms of polarity. It would be nice to have all that printed onto a paper page; however, the print function of that software only prints what you can see on the screen. Zooming out to get the entire sampled interval did not help. That made it impossible to see the signal details.

I just saved the captured data as a text file to end up with a 25-MB file from a 20-s capture at a 20-µs sample rate. A quick look at the file with Notepad showed that lines starting with a space contained data arranged as three columns: sample time, channel A, and channel B. I connected the RECEIVE signal to channel A and the transmit signal to channel B.

The next step was to figure out a good way to turn this data into a really long graph that could be printed. Because samples were numeric values expressed in millivolts, I used a suitable threshold to convert each sampled value to a digital value of 1 or 0.

VECTOR GRAPHICS

Some of the most popular graphics formats like BMP, JPEG, and GIF are bitmap graphics. They represent the image matrix of picture elements (pixels), each one a given color. There are different code libraries for you to read and write such file formats. However, I was more inclined for a solution that was as self-contained as possible. Because I was using Java, interfacing with an external library was not easy.

Vector graphics create a representation of the drawing actions, instead of the color of each pixel. It is like the set of commands to create the image. There are different file formats for vector graphics. Most of them are proprietary, but the World Wide Web Consortium developed the SVG mark-

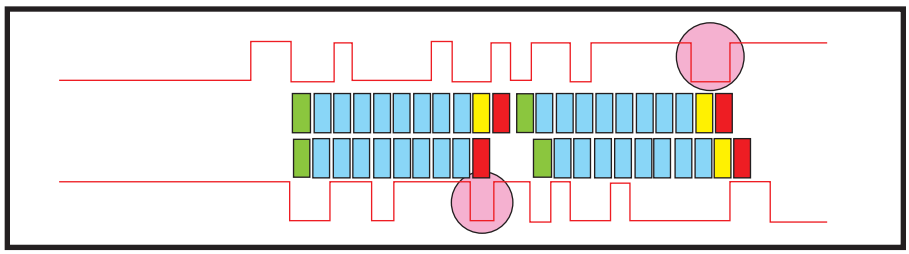


Figure 2—*The top and bottom signals are the same but inverted. The circles show when there is a violation of the stop bit.*

GX typhoon

infinite possibilities



:: Expansion port for optional fan control
:: MCE support for visual feedback
:: Simple USB connection with all software
:: Display system information
:: Integrated remote

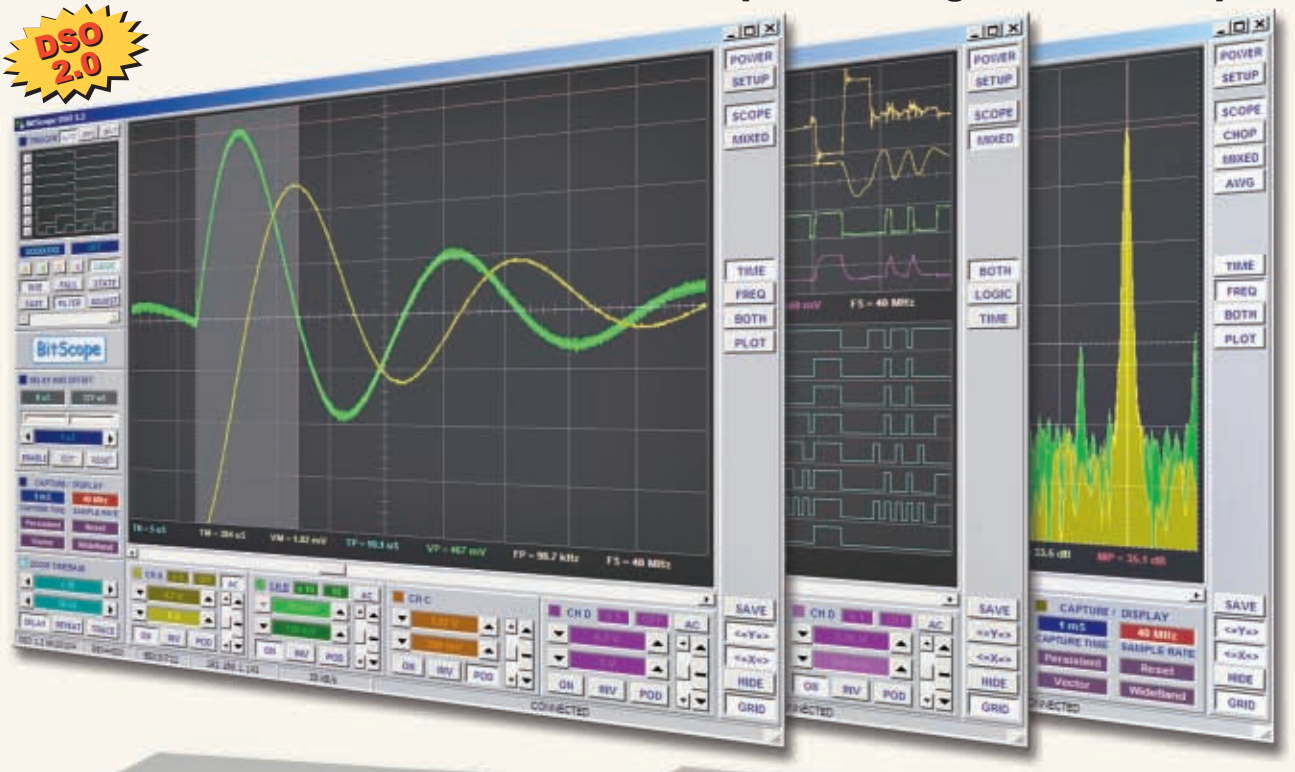
WWW.MATRIXORBITAL.COM

MATRIX ORBITAL

BitScope PC Oscilloscopes & Analyzers

DSO Test Instrument Software for BitScope Mixed Signal Oscilloscopes

DSO 2.0





4 Channel BitScope



2 Channel BitScope



Pocket Analyzer

Digital Storage Oscilloscope
✓ **Up to 4 analog channels using industry standard probes or POD connected analog inputs.**

Mixed Signal Oscilloscope
✓ **Capture and display up to 4 analog and 8 logic channels with sophisticated cross-triggers.**

Spectrum Analyzer
✓ **Integrated real-time spectrum analyzer for each analog channel with concurrent waveform display.**

Logic Analyzer
✓ **8 logic, External Trigger and special purpose inputs to capture digital signals down to 25nS.**

Data Recorder
✓ **Record anything DSO can capture. Supports live data replay and display export.**

Networking
✓ **Flexible network connectivity supporting multi-scope operation, remote monitoring and data acquisition.**

Data Export
✓ **Export data with DSO using portable CSV files or use libraries to build custom BitScope solutions.**

BitScope DSO Software for Windows and Linux

BitScope DSO is fast and intuitive multi-channel test and measurement software for your PC or notebook. Whether it's a digital scope, spectrum analyzer, mixed signal scope, logic analyzer, waveform generator or data recorder, BitScope DSO supports them all.

Capture deep buffer one-shots or display waveforms live just like an analog scope. Comprehensive test instrument integration means you can view the same data in different ways simultaneously at the click of a button.

DSO may even be used stand-alone to share data with colleagues, students or customers. Waveforms may be exported as portable image files or live captures replayed on other PCs as if a BitScope was locally connected.

BitScope DSO supports all current BitScope models, auto-configures when it connects and can manage multiple BitScopes concurrently. No manual setup is normally required. Data export is available for use with third party software tools and BitScope's networked data acquisition capabilities are fully supported.





www.bitscope.com

up language standard in 1998 to provide a standardized alternative. I had learned about SVG several years ago, but I was never interested in that file format before. One year ago, I started using Inkscape graphics software and it turns out that it uses the SVG file format to save its designs. SVG files are human-readable files like HTML (although you may prefer a browser program to do the processing). For example, the following command will create a diagonal red line: `<line x1="5" y1="5" x2="45" y2="45" stroke="red"/>`. There are several other commands like circle, polyline, polygon, and ellipse.

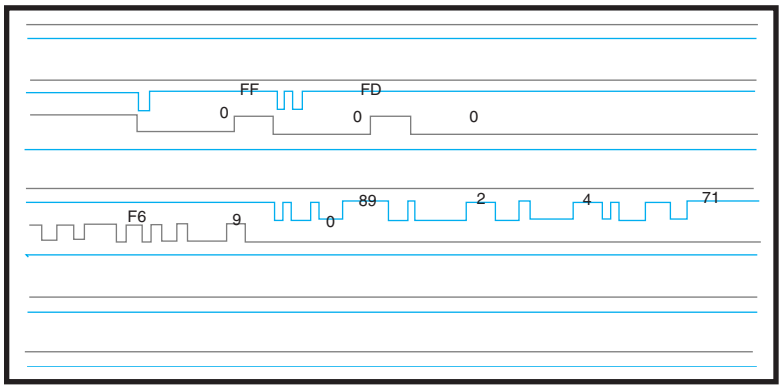


Figure 3—*The decode.java program adds the decoded values to the drawing.*

It is quite simple for a program to print the sequence of commands to create the graphical representation of a digitized signal. In my case, I wrote some code to draw the two signals I had previously digitized, as if they were the lines of a book. Since page width is limited, I drew from left to right, creating a line and then moved down a bit to start from the left of the next fragment of the signal. I repeated the process until all the captured data was drawn. A different color for each signal helped me differentiate them. I wrote the Stat.java program to get an idea of the maximum, average, and minimum values of the captured data and to solve the details about properly parsing the data file. For some weird reason, the data file contained both commas and dots as decimal separators used interchangeably, so I took care of that before converting the data into a number. I guess this problem might have been caused by a computer's regional configuration.

Draw.java created an SVG file representing the captured data as a set of lines using two colors (blue and red) to represent channels A and B. This way you can print it out or study the signal on-screen while viewing several seconds of captured signals. Then it is much easier to look for patterns of activity. I used free Inkscape software to show the results (see Photo 3).

GETTING THE BYTES OUT

I did several captures using different address codes (2, 7, 9, 15, and 18) in the keyboard of a user typing the code 1,2,3, and 4. A sequence of events associated with each keystroke became apparent (see Figure 1). The bus transmit signal, commanded by the alarm motherboard, appears as a red line. After a long pulse, much longer than a character, the keyboard starts transmitting some pulses (a blue line) triggered by the fall edges of the transmit signal. I guessed this was some kind of polling action. Later, after about a 50-ms pause, the transmit signal transmitted some bytes and then the keyboard panel transmitted what appeared as a keystroke message. Finally, the bus transmit signal sent what might be an acknowledgment of the keystroke message.

It was not clear what type of transmission was used or the polarity of the signals. A standard RS-232 data signal stays at negative voltage while idle, which is the opposite polarity of the logical level high (positive voltage) it represents. An asynchronous transmission will start with a 0 bit (start bit), but the line will remain high until then.

Here you see a signal that mostly remains at a low level (red) and another one that mostly remains at a high level (blue). At first sight, it might look like we need to invert the transmit signal (red). However, if we invert the red signal, then the last line does seem to break because characters do not seem to fit to either 8 data bits without parity or to 8 data bits plus a parity bit.

Figure 2 shows the bus-transmit signal (below) and the same signal inverted (above) against a data format of 8 bits, no parity (below), and 8 data bits plus 1 parity bit (above). I have located the patterns starting at the first falling edge of the signal (that it marks the beginning of a start bit). It is expected that after transmitting the data bits (and maybe parity) there has to be a stop bit (a red block in the figure) that has to be a 1 bit. The signal on the bottom of the figure is the only one that it is consistent with an asynchronous transmission of 8 data bits plus parity bits. So the transmit signal has the right logic polarity but it violates the idle condition, so when no data is being sent, the line is kept low and it is raised to the high level some milliseconds before the beginning of a data transmission or as a polling action (line high for more than some milliseconds without transmitting data). If a poll is detected, the keyboard will initiate its own addressing phase.

We are almost done, but some understanding of the message's content is still required. To do this, I needed to simulate a UART, because it is a pain to manually decode each byte from the already generated graphs. Because the capture sampling rate was approximately 10 times the bit time, I used this information in the decoding routine. SVG format allows text labels on the document, so I added them in the last Java program: Decode.java. This program creates an SVG file,

Address	Values (hexadecimal)	Binary (least significant bit first)
2	FB	1101 1111
7	7F	1111 1110
9	FF, FD	1111 1111 1011 1111
18	FF, FF, FB	1111 1111 1111 1111 1101 1111

Table 1—*Want to know how addresses are made? It is all clear when you look at the binary pattern.*

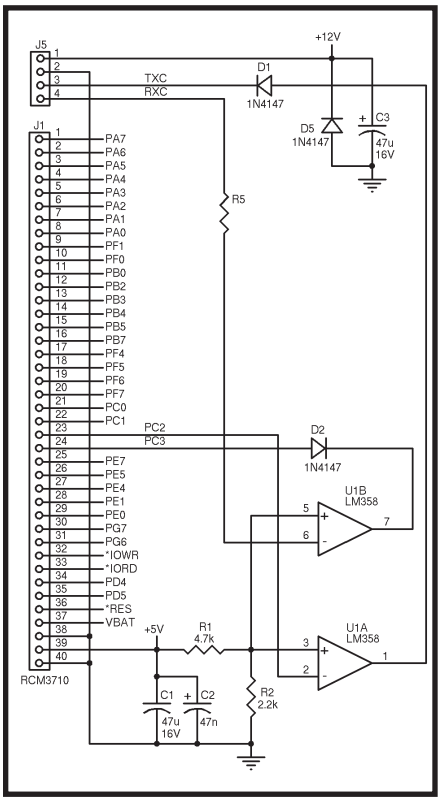


Figure 4—*I connected a Rabbit core to the ECP bus.*

drawing the signals and adding the values of each decoded byte as a hexadecimal value.

Once the data decoding is printed over the graph, it is much simpler and convenient to figure things out. So, the addressing scheme shows differences depending on the address value used. In Figure 3, address nine turns into the sequence FF, FD. Table 1 shows you other address values too. You just need to put these values in binary form (using the serial-transmission order of sending the least significant bit first) to see how addressing works in the bus. (The address is the number of consecutive bits before a zero is found. Multiple bytes are used as needed.) A "GO AHEAD" message is formed by a fixed F6 data byte plus the address number in the next byte. Each keystroke message is preceded by 2 bytes, then the actual number pressed represented in binary (not ASCII), and then a fourth byte (a checksum byte). Acknowledgment is a single byte matching the first byte of the keystroke message. If that is missing, the keyboard will attempt to retransmit the message when the next poll happens.

Observing different keystroke messages allowed me to discover the details of the first byte of the message and the checksum calculation. Messages also include a sequence number to detect duplicated messages that can happen in case of a transmission error. The second byte was just a counter of the number of bytes to follow, but it was always two unless I typed really fast and then two different keystrokes would be sent in a single message, but 1 byte longer.

TRYING THINGS OUT

My goal was to be able to simulate keystrokes on the ECP bus. I thought I knew how everything worked, but you are never sure until you can fool the real system. So, I decided I needed to check if my understanding of the protocol was good enough. The simplest way of getting things done was to use a PC serial port to simulate the complete dialog. But there were one or two problems. I was not sure if I could get good time measurements in Java and I had never used Java for serial port programming.

After searching the Internet for a minute, I decided to use Perl for this task. I knew I could use the "`select`" system call to get quite accurate time delays. I could also transmit and receive over the serial port. Once my code was working properly, I learned that my guesswork was a success. With the inverted nature of the bus signals, I just needed a diode to connect to the computer. I connected the bus transmit signal to both the RXD and CTS signals. I connected the bus receive to the TXD signal of the PC serial port.

I have to admit that results were not perfect due to the fact that millisecond precision under Windows is not always achieved (at least not with my Perl code). For my second attempt, I used a Rabbit Semiconductor RCM3710 core. I used a small circuit to invert and adapt the signal levels. A couple of op-amps were used in this case, as shown in Figure 4. The logic of keypress.c code is the same as Perl code, but here the timing is just perfect because there are no other tasks to interfere with.

While I am not providing the complete details of how the protocol works in the text, you will find them in the article's source code. I hope that you might consider using the SVG file format in your future projects when you need to create some graphics from your programs. If you do, you will find that Inkscape software will help you display, modify, and print your graphs. The drawing and decoding code I provided here may be adapted for a similar project, but using a different data rate or format. I will be glad if you can use it and I hope you will be inspired the next time you have to deal with an undocumented bus.

Miguel Sanchez holds a B.S., an M.S., and a Ph.D. in Computer Science. He has taught computer networks courses at the Technical University of Valencia, Spain since 1989. Miguel's interest in electronics and microprocessors sparked his career in computer science, but his solder is always at hand. His research is currently focused on vehicle energy efficiency. He also works as a consultant. You may contact him at misan@disca.upv.es.

PROJECT FILES

To download code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/201.

RESOURCES

CPAN Search, "Win32::SerialPort Perl package," http://search.cpan.org/dist/Win32-SerialPort/.

Inkscape, "SVG Graphics Editor," www.inkscape.org.

Kevin Lindsey Software Development, "Basic SVG shapes," 2003, www.kevlindev.com/tutorials/basics/shapes/svg/index.htm.

SOURCES

DS1M12 Stingray oscilloscope
USB Instruments
www.usb-instruments.com

RCM3710 Core
Rabbit Semiconductor
www.rabbitsemiconductor.com

Uncomplicated RF Communication

You don't have to build or run an official ZigBee stack to use the new 900-MHz and 2.4-GHz transceiver ICs. However, knowing what the radio is capable of and how to use its features will enable you to easily understand what a ZigBee stack is doing. This month, Fred teaches you all about ZigBee stacks.

It's rather easy to get caught up in the hype and find yourself standing lost amongst the trees in the forest. One of today's biggest hypes is ZigBee. Don't get me wrong, I'm not bashing ZigBee. I'm simply bringing to your attention the amount of interest ZigBee is getting these days.

When I initially signed up for ZigBee school, I was totally overwhelmed by the complexity of the ZigBee specification. I was equally befuddled by the seemingly complex RF side of ZigBee. What I should have done was back off of the high-level ZigBee stuff and establish a groundplane understanding of the hardware that makes ZigBee possible. The hard fact is that ZigBee is simply an extension of the IEEE 802.15.4-2003 specification, and the specification can't exist without low-power 900-MHz and 2.4-GHz transceiver hardware.

An elegantly written ZigBee stack is a beautiful thing. However, the most complicated and feature-rich of ZigBee stacks still has to cater to the RF hardware that supports it.

What I'm driving at here is that you don't have to build and run an official ZigBee stack to use the current crop of 900-MHz and 2.4-GHz transceiver ICs. However, knowing what the radio is capable of and how to access its capabilities makes for an easier understanding of what a ZigBee stack is doing, if your application requires a stack.

A couple of years ago, we cracked the hard outer shell of the 802.11b nut. If you recall, we didn't have very much 802.11b technical information to support us. The lack of technical documentation is not a factor in the IEEE 802.15.4 world, where datasheets and reference design Gerber files are standard handouts. One of the most prolific 2.4-GHz transceiver ICs out there right now is the Texas Instruments CC2420. If you come to understand the CC2420, you will find that many of its competitors walk in the trails blazed by its tank tracks.

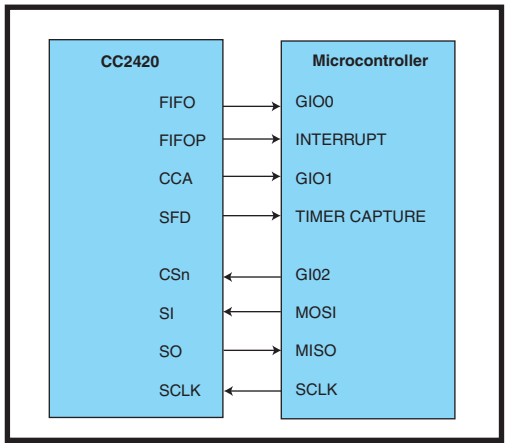


Figure 1—*This represents a typical interconnect between the CC2420's digital I/O pins and a generic host microcontroller. Note that although the physical connection is there, the CC2420 driver firmware does not employ the services of the CC2420's CCA pin.*

CC2420 OVERVIEW

Like many of the IEEE 802.15.4-compliant transceiver ICs, the CC2420 uses SPI to communicate with a host, which is usually a microcontroller equipped with its own integrated hardware SPI interface. The whole idea behind IEEE 802.15.4 is low-power (20 mA) operation coupled with low data rates (250 KBps). The CC2420 was designed with all of that in mind. Direct-sequence spread spectrum (DSSS) is another technique you will find that is common to radios, aimed at IEEE 802.15.4 use within the industrial, scientific, and medical (ISM) frequency domain. Naturally, the CC2420 includes a digital DSSS baseband modem that provides a spreading gain of 9 dB. The popularity of the CC2420 stems from its international flavor. The CC2420 complies with all of the regulations put forth by the United States (FCC CFR47 Part 15), Europe (ETSI EN 300 328 and EN 300 440 class 2), and Japan (ARIB STD-T66).

The CC2420 is not just a means of temporarily bending the Earth's magnetic field. Lots of things are done within the CC2420 to reduce the workload of the host microcon-

```
Register Read
0 1 A5 A4 A3 A2 A1 A0 → 16 bits of data returned D15–D0

Register Write
0 0 A5 A4 A3 A2 A1 A0 D15...D0
```

Figure 2—*This is typical of a SPI command/data transfer sequence. You've got all of the essential ingredients: a command, an address, and data.*

XOSC16M_STABLE	Bit 6
TX_UNDERFLOW	Bit 5
ENC_BUSY	Bit 4
TX_ACTIVE	Bit 3
LOCK	Bit 2
RSSI_VALID	Bit 1

Table 1—*Short and sweet. One look is all it takes to get a complete status.*

troller. Packet-handling chores, such as data buffering and data authentication, are handled by the internals of the CC2420. The CC2420 can also assist in the transmission and reception phases of operation by performing clear channel assessment (CCA) and providing link quality information (LQI).

ACCESS THE CC2420 INTERNALS

Everything you will do with the CC2420 will depend heavily on the SPI connection between the host microcontroller and the CC2420. The CC2420 is always configured as a SPI slave and connects to the host using a standard SPI hookup like the one shown in Figure 1.

The CC2420's register farm is composed of 33 16-bit configuration and status registers, 15 command-strobe registers, and a pair of 8-bit registers that act as portals to the independent transmit and receive FIFOs. The addressing scheme associated with reading and writing the CC2420's set of 50 registers is very efficient. Each CC2420 register is addressed with 6 bits. The value of the most significant bit of the address byte determines if a `RAM(1)` or a `Register(0)` operation is about to take place. Bit 6 of the address byte, acts as the `Read/Write(1/0)` signal. The actual 6-bit register address begins with bit 5, which is the most significant bit of the 6-bit address value. In the case of the 33 16-bit configuration and status registers, 24 bits are clocked across the SPI for any read or write operation when the CC2420's chip select (CSN) pin is held logically low. Figure 2 is what the Read/Write SPI binary bit stream looks like.

Accessing the CC2420's 15 command-strobe registers is akin to issuing a single-byte write instruction to the CC2420. Command-strobe registers are used to do things like enable the crystal oscillator or flush the transmit FIFO. There's also a no operation (SNOP) command-strobe register that allows reading of the status bits by way of the SPI. A CC2420 status byte is returned on the CC2420's SO pin when a command strobe is issued and during a transfer of the register-access byte. CC2420 status bytes are also clocked onto the CC2420's SO pin, while clocking the first RAM address byte and during a transfer to the CC2420's TXFIFO. The status byte's most significant bit and least significant bit are reserved, leaving only 6 bits of status information (see Table 1).

Bank 0 TXFIFO	0xb00
Bank 1 RXFIFO	0xb01
Bank 2 Security	0xb10

Table 2—*Absolutely no rocket science here.*

Bit six of the CC2420's status byte informs you about the state of the CC2420's 16-MHz oscillator. If a FIFO underflow occurs during a transmission, the `TX_UNDERFLOW` bit is set. Bit 5 must be cleared using a `SFLUSHTX` command strobe. The ENC in bit 4 of the CC2420 status byte is short for encryption. The CC2420 has an internal encryption module if you care to use it. The only remaining CC2420 status bit that may not be obvious as to its usage is the LOCK bit, which simply reports if the CC2420's PLL is locked or not. The collection and interpretation of the CC2420 status byte is very important.

All but one command strobe will not be accepted when the CC2420's crystal oscillator is not running. Only the `SXOSCON` command strobe will be recognized by the CC2420 when the CC2420's crystal oscillator is inactive. The remaining CC2420 command strobes are accepted only after the `XOSC16M_STABLE` status bit is set.

While you're on the subject of command strobes, a command-strobe register is accessed by issuing a register-write operation. No data is written to the command-strobe register. Only the RAM/Register bit, the Read/Write

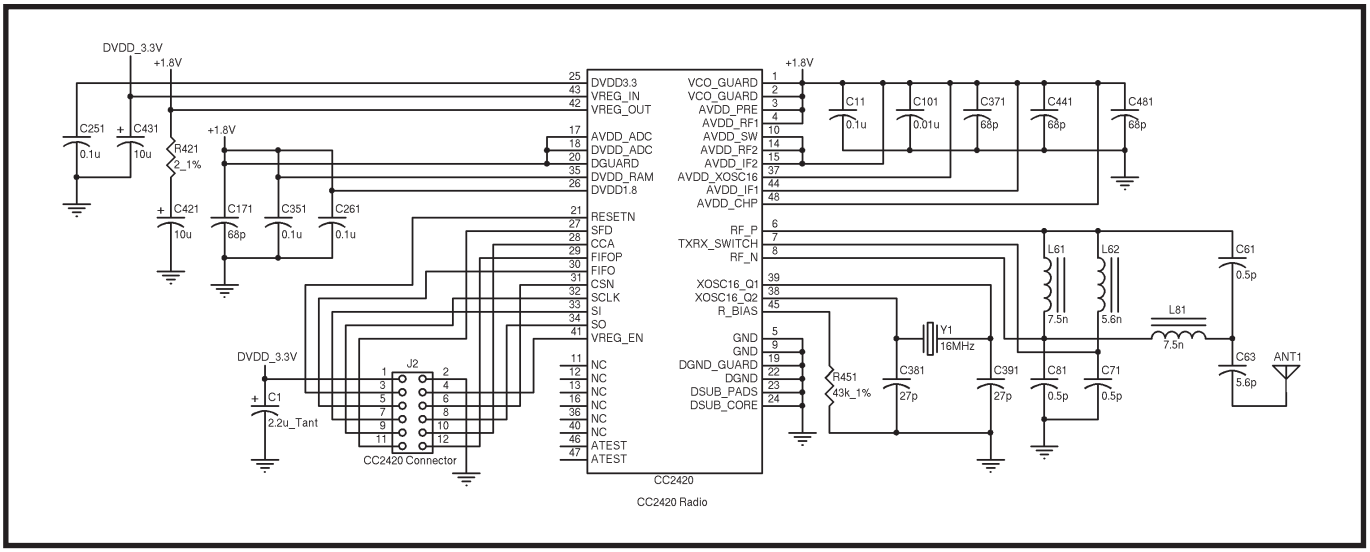


Figure 3—*This is textbook CC2420 implementation. Texas Instruments doesn't recommend you deviate too much from this design and the layout you see in Photo 1.*

bit (both set to 0), and the 6-bit command strobe-register address values are transferred on the SPI.

The CC2420 houses 368 bytes of RAM in addition to its 50 configuration and status registers. Only a single byte of RAM data can be read or written at a time. The CC2420 programmer must submit only the beginning address when initiating a read or write because the CC2420 hardware will automatically increment the address with successive RAM read/write operations.

The RAM address consists of 2 bytes. The most significant bit of the most significant byte is the RAM/Register bit, which must be set logically high to access the CC2420's RAM area. The 7-bit RAM address follows the RAM/Register bit. The CC2420's RAM area is partitioned into three banks (see Table 2).

The second address byte begins with the binary bank value followed by the Read/Write bit. The rest of the second address byte is ignored by the CC2420. From the SPI's point of view, the RAM addressing scheme looks like this:

```
1 A6 A5 A4 A3 A2 A1 A0 B1 B0
  RW X  X  X  X  X
```

It is not recommended to do direct reads and writes to the FIFO RAM area because the automatic FIFO counter will not be in the loop. The TXFIFO and RXFIFO RAM areas can be accessed instead through the TXFIFO and RXFIFO registers. FIFO access using the TXFIFO and RXFIFO registers is byte-by-byte. Issuing a `SFLUSHTX` command strobe will clear the TXFIFO, while an `SFLUSHRX` (issued twice by recommendation of the CC2420 datasheet) will flush the RXFIFO.

CC2420 RECEIVE MODE

Now that you know how to access the CC2420's internal registers and RAM areas, there should be no problem coding up the necessary initialization routine. However, we still need to obtain more information before attempting to code the receive and transmit routines. The CC2420 uses three I/O pins to assist in the receive process: SFD, FIFO, and FIFOP.



Photo 1—*You can't get this CC2420 radio commercially anymore. But, if you're resourceful, you can build your own. The specifications and PCB files are available on the Texas Instruments web site. You can also get CC2420 reference design evaluation boards from Texas Instruments in its CC2420 development kits.*

A complete RF transmission consists of a preamble (4 bytes of 0x00) followed by a start of frame delimiter (SFD), 0xA7 the length (least sig-



PC/104 Single Board Computers

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



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

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

200 MHz ARM9
Power as low as 1/4 Watt

- 5 boards, over 2000 configurations — $99 qty 100
- Fanless, no heat sink
- SDRAM - up to 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
- VGA video
- Linux, Real Time extension, NetBSD

NEW!
- SD card option

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



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

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

133 MHz 586

- Power as low as 800mA — $229 qty 100
- Fanless, no heat sink
- SDRAM - up to 64MB — $259 qty 1
- COM Ports - up to 4 ports
- Ethernet Ports
- DIO Channels - up to 40
- PCMCIA II adaptor
- Compact Flash adaptor
- USB Ports (Except on TS-5300)

see our website for 33 MHz 386 configurations

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

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

New Products and PC/104 Peripherals

Tiny WiFi Controller boots Linux in 1.1 seconds



$249 qty 1

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

Intelligent Battery Back-up

NEW!

$119 qty 1

- Run your system for days with no external power source



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

see our website for more boards and option details

Technologic SYSTEMS

We use our stuff.

Visit our TS-7200 powered website at

www.embeddedARM.com

Every week, Jameco is giving extra discounts on major brands like these:

- AlcoSwitch • AMP/Tyco
- Dallas Semiconductor
- Intersil • Maxim
- Microchip • Molex
- Panasonic • AVX
- Fairchild Semiconductor
- Tyco • Aromat
- Aavid Thermalloy
- Atmel • ST Micro

WIN BIG



- T.I. Semiconductor
- Augat/Tyco • Bourns
- CTS • Cypress
- C&K Switches
- Comair Rotron
- Condor • Grayhill
- Intel • Micron
- Philips • Power-One
- Raychem/Tyco
- Renesas • SanDisk
- Toshiba
- Vishay Siliconix

EVERY WEEK AT JAMECO

TAKE AN EXTRA 15% OFF ON 2 NEW MAJOR BRANDS EVERY WEEK!

And that's in addition to the industry-best pricing you already get at Jameco every day!

You already know about Jameco's best-in-the-business low pricing...

Now we're out to get a *new* message across: *Major Brands—at Jameco pricing!*

To dramatize the point, we're giving you an *additional* 15% discount on two different major brands each week. From now through the end of May!

SEE WHO'S UP THIS WEEK GO TO THE WEBSITE

And that's on *top* of our everyday discounted pricing!

We wanted to make this so good a deal that you just *couldn't* pass it up.

So... Want to know which two major brands you can save extra-big on *this* week? Just head to the website...

www.Jameco.com/CCW

Great Products. Awesome Prices.

JAMECO®

nificant 7 bits of 1 byte) of the MAC protocol data unit (MPDU), and the MPDU itself. The CC2420's SFD pin will go logically high upon the detection of a valid IEEE 802.15.4 SFD sequence. The SFD pin will go low again after the reception of the last byte of the MPDU. The SFD pin will also go low immediately if address recognition fails.

To aid in identifying an address recognition failure, the CC2420 employs the services of the FIFO pin. The FIFO pin will go logically high following the reception of the length byte, which will be the first byte stored in the RXFIFO. If address recognition fails, both the SFD and FIFO pins will go logically low. As long as the FIFO pin remains logically high, there are data bytes in the RXFIFO. The FIFO pin will take a logically low state when the RXFIFO is empty. Thus, when address recognition completes successfully, the FIFO pin will remain high as the SFD pin goes low to indicate the end of MPDU reception.

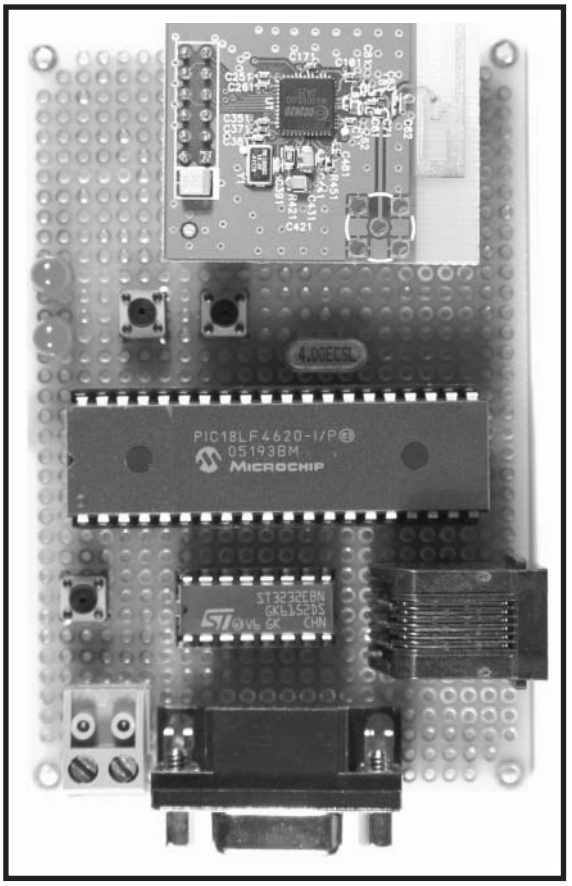


Photo 2—*It doesn't take much computer power to put a CC2420 radio on the air. In many ways, using a PIC18LF4620 here is overkill. As usual, I used 0805 SMT passives on the other side of this point-to-point wired board.*

Another CC2420 I/O pin, FIFOP, is also used to verify MPDU reception. The FIFOP pin will go high when the number of unread bytes in the RXFIFO exceeds the threshold programmed into the 7 FIFOP_THR bits of the IOCFG0 register. Also, the FIFOP pin will not go high until address recognition completes successfully, even if the number of bytes in the RXFIFO exceeds the programmed threshold. A successful MPDU reception is signaled by the SFD pin going low and the FIFOP line going high, as the FIFOP line will go logically high and the SFD pin will take on a logical low when the last byte of the MPDU is received, regardless of the RXFIFO threshold.

Figure 1 should be a bit clearer to you now. In a standard IEEE 802.15.4-compliant configuration, and provided the FIFOP threshold value is set to 127 (the maximum value of the length byte), the FIFOP pin can be used to signal the receipt of a complete frame by interrupting the host microcontroller. If multiple frames are buffered in the RXFIFO, the FIFO pin can be used to interrogate the RXFIFO for the presence of unread bytes. And, if frame timing is important to you, the timing window created by the SFD pulse width can be captured by a host microcontroller capture input.

CC2420 TRANSMIT MODE

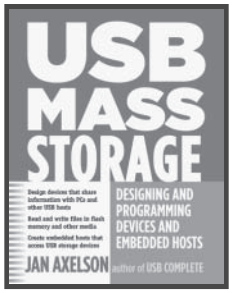
We can take or leave the functionality offered to us by the clear channel assessment (CCA) pin. The CCA signal is based on the measured receive signal strength indicator (RSSI) and a programmable threshold value located in the RSSI and CCA status and control register. The CCA function is used to implement the carrier sense multiple access with collision avoidance (CSMA-CA) functionality. Issuing the STXONCCA strobe command is the easiest way to kick-off the CSMA-CA process. By issuing a STXONCCA strobe command instead of the STXON strobe command, transmission will only occur if the channel is clear. The TX_ACTIVE status bit can then be used to detect the result of the CCA instead of reading the logic level of the CCA pin. CCA has three modes, which are mutually exclusive and determined by a register setting. Mode 1 clears the channel when received energy is below the threshold. Mode 2 clears the channel when it is not receiving valid IEEE 802.15.4 data. Mode 3 clears the channel when energy is below the threshold and it is not receiving valid IEEE 802.15.4 data.

The FIFO and FIFOP pins are not considered during Transmit mode because they only relate to the RXFIFO. However, the SFD pin is active during Transmit mode and goes logically high when the SFD field has been completely transmitted. When the number of bytes designated by the length byte have been transmitted, the MPDU is considered to be fully transmitted and the SFD goes logically low at that time. The CC2420's SFD pin will also take on a logically low level on a TXFIFO underflow. The TXFIFO underflow condition is verified by reading the state of the TX_UNDERFLOW status bit.

PLINKING A CC2420 MESSAGE

The CC2420 is remarkable, but it needs a supporting cast of its own in addition to the host microcontroller circuitry. Figure 3 is a graphic representation of the components necessary to immediately support the CC2420. The circuit you see can be assembled on a small double-sided PCB. The CC2420 reference design PCB art and Gerber files can be obtained from Texas Instruments. As you can see in Photo 1, I "bogarted" my CC2420 radio and supporting electronics from an old Microchip Technology PICDEM Z development kit. As you know, Microchip now ships its own

ADD MASS STORAGE
TO YOUR DESIGNS



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

USB/ETHERNET DAQ
LabJack
UE9

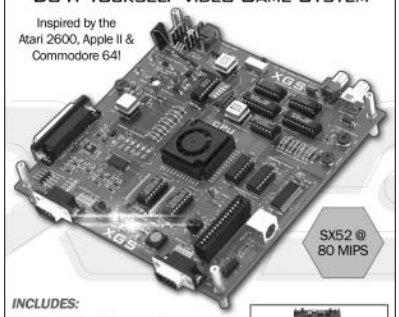


Available now
for only ...
$429 qty 1
USB/Ethernet Data
Acquisition & Control

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

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

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



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





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

Listing 1—*This is the main function of the test version of my CC2420 driver firmware. The real meat of the CC2420 driver lies in the supporting functions that fill and empty the CC2420's internal registers and RAM areas.*

```
void main(void)void main(void)
{
        unsigned int timer,x;

        CC2420_Enable();
        sequence_byte = 0x00;
        do{
        CC2420_CSn_0();
        SPI_OUT(REG_TXFIFO);
        SPI_OUT(0x1A); //length (not counting this byte)

        SPI_OUT(0x01); //frame control LSB
        SPI_OUT(0x88); //frame control MSB

        SPI_OUT(sequence_byte++);     //sequence byte

        SPI_OUT(0xAA); //destination PAN ID LSB
        SPI_OUT(0xAA); //destination PAN ID MSB

        SPI_OUT(0xBB); //destination addr LSB
        SPI_OUT(0xBB); //destination addr MSB

        SPI_OUT(0xCC); //source PAN ID LSB
        SPI_OUT(0xCC); //source PAN ID MSB

        SPI_OUT(0xDD); //source addr LSB
        SPI_OUT(0xDD); //source addr MSB

        SPI_OUT('C');  //data
        SPI_OUT('I');
        SPI_OUT('R');
        SPI_OUT('C');
        SPI_OUT('U');
        SPI_OUT('T');
        SPI_OUT(0x20);
        SPI_OUT('C');
        SPI_OUT('E');
        SPI_OUT('L');
        SPI_OUT('L');
        SPI_OUT('A');
        SPI_OUT('R');
        CC2420_CSn_1();
        Transmit_Frame();
        for(timer=0;timer<0xFFFF;++timer)
        {
                ++x;

        }
        }while(1);
}
```

MRF24J40 radio card with the new PICDEM Z development kits.

The host microcontroller board, shown in Photo 2, is based on a Microchip Technology PIC18LF4620 and wired in accordance with Figure 4. Most of the work is done in firmware. All you really need the PIC for, in addition to its program execution core, is its hardware SPI port and interrupt capabilities.

The firmware needed to drive the

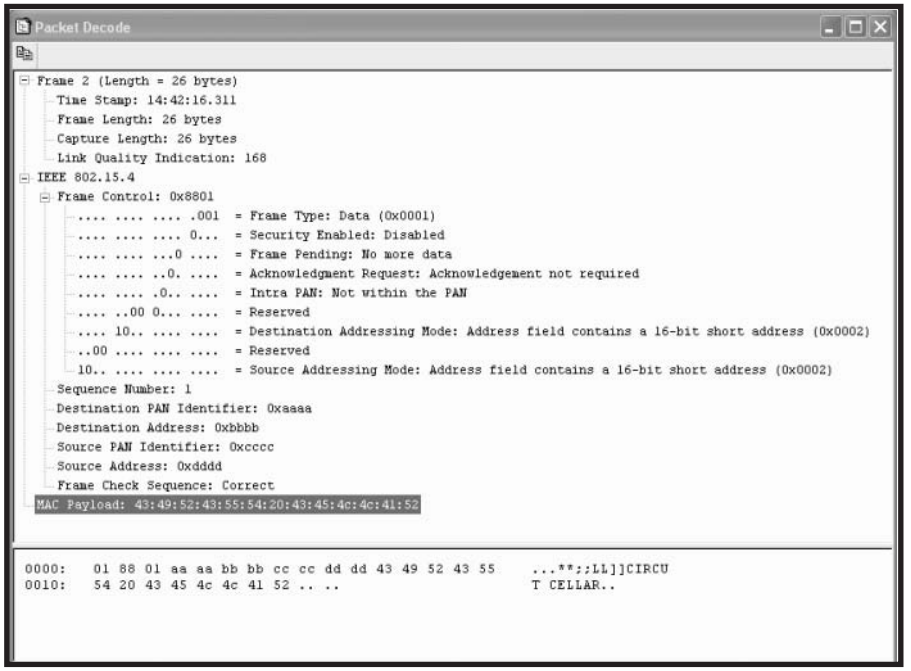


Photo 3—*I am able to do lots of "what ifs" with the CC2420 firmware by using Daintree Networks's SNA Sniffer as a catcher's mitt. I used this shot to verify the correct transmission of my hand-made IEEE 802.15.4-compliant frame.*

CC2420 is not complex, but it is a bit more than I can show in print form right now. So, I posted the HI-TECH Software PICC-18 C source code on the *Circuit Cellar* FTP site. Let's take a look at what happens within the hardware initialization portion of the CC2420 driver firmware. First, the PIC18LF4620's CC2420 general-purpose I/O pins are initialized. Then the PIC18LF4620's SPI is initialized. The PIC18LF4620's hardware initialization processes consist of configuring general-purpose I/O pin directions (input/output) and setting their initial states, which put the CC2420 into an unselected and powered-down state. Once the PIC18LF4620 is configured, you can turn your attention to initializing the CC2420's PHY. That entails several steps (see Figure 5).

The remaining steps that pertain to getting the CC2420 ready to receive and transmit involve selecting a transmit power level and flushing both the RXFIFO and TXFIFO. The transmit power level is selected according to how much current you wish to add to your overall current-consumption budget. In the course of all of this action, the FIFOP threshold is set to 127 and the CC2420 MAC security feature is disabled.

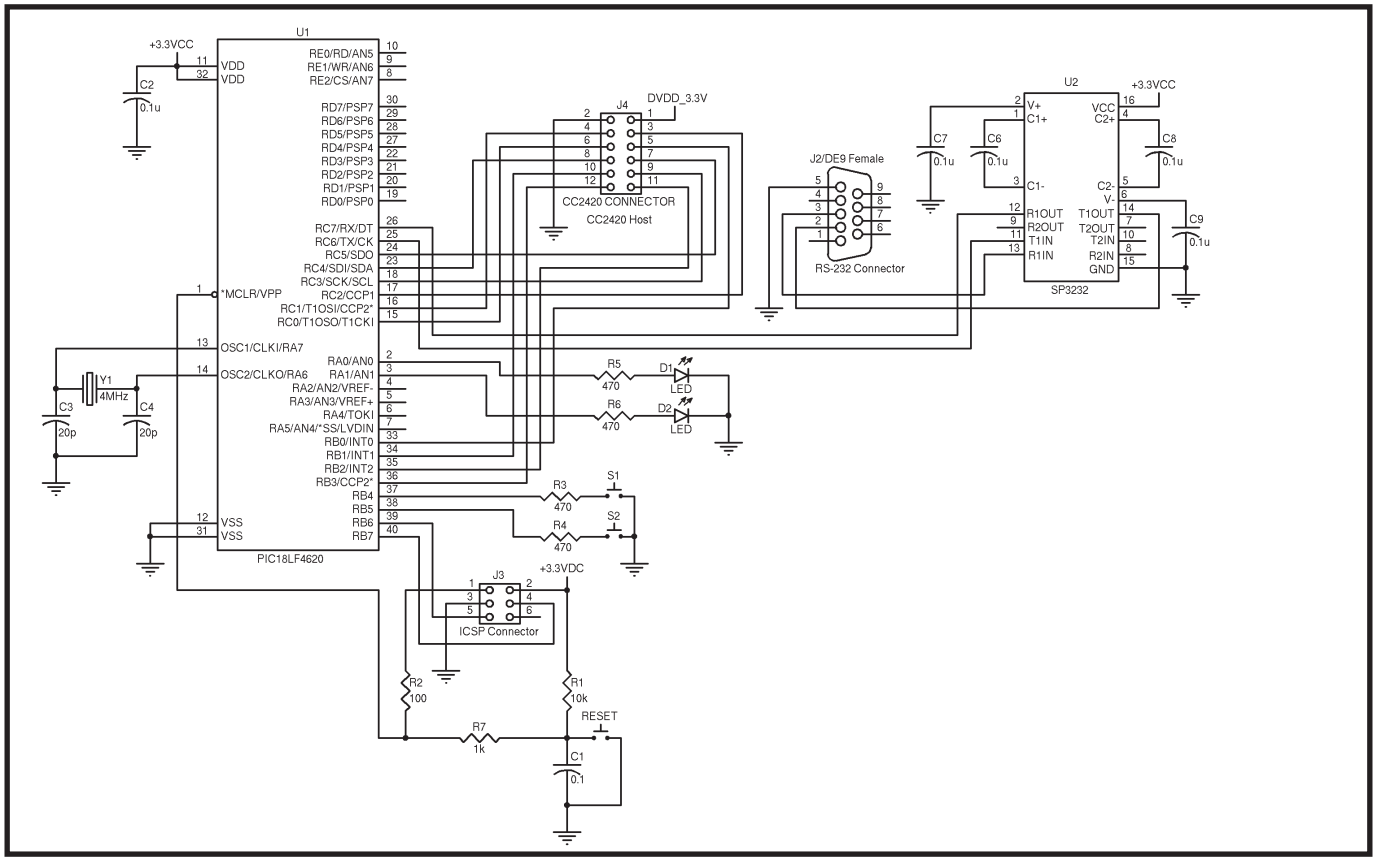


Figure 4—*Any microcontroller with hardware-SPI capability and a good interrupt engine can be inserted into this picture. All you really need is to be able to communicate with the CC2420 via a SPI and have the ability to manipulate the CC2420's internal registers and RAM areas.*

Give the Customer More!

If your company has a physical storefront, you may qualify to become a *Circuit Cellar* distributor.

Circuit Cellar will:

- Entice customers to stay in your store longer
- Attract repeat walk-in business
- Serve as a useful product demonstration tool

Contact Sean Donnelly
sean@circuitcellar.com

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

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

1. Issuing a hard CC2420 reset-pin-associated reset
2. Enabling the CC2420's internal 1.8-V regulator
3. Enable the CC2420's 16-MHz oscillator via an SXOSCON command strobe
4. Polling the status byte until X0SC16M_STABLE is set
5. Set modem control registers to:
   a. Enable address decoding
   b. Set CCA mode to 1
   c. Enable automatic CRC
   d. Enable automatic ACK generation
   e. Set the preamble length for IEEE compliancy

Figure 5—*This is standard stuff. The PHY is very important. Everything passes through the PHY.*

The next step in the CC2420 initialization process involves loading address information into the CC2420's address registers. This is done to ensure that the CC2420's address recognition engine has valid addressing information to work with. CC2420 RAM address space is reserved for addresses (16 bits), a PAN identifier (16-bit address), and a 64-bit IEEE address. The addressing information is sent to each of the address-bearing CC2420 RAM locations with simple SPI transfers.

The operating channel (11) is loaded into the `FSCTRL` register just before the CC2420 driver firmware sets up the capture interrupt, which is multiplexed to general-purpose I/O RB3, and configures the PIC18LF4620 for PORT B interrupt on change. The final initialization step enables all of the PIC18LF4620 interrupts.

You really need to make sure you're sending what you think you're sending before trying to receive. So, instead of getting fancy, I kept low to the ground and chinked out the code you see in Listing 1. All of the CC2420 and PIC18LF4620 hardware initialization code is contained within the braces of the `CC2420_Enable` function.

Rather than bore you with play-by-play rhetoric, I put my money where my mouth was and produced the Daintree Networks's SNA Sniffer shot in Photo 3 from the code snippet you see in Listing 1. I basically hand-assembled an IEEE 802.15.4-2003 data frame, loaded it into the CC2420's TXFIFO, and initiated a transmit with an `STXONCCA` command strobe from within the `Transmit_Frame` function. Since this is a test shot, I decided to demark all of the address fields so you can see how they would be laid out in the transmission of the frame.

IT TAKES TWO TO TANGO

All that's left is to test the receive firmware, which is nothing more than an interrupt handler that employs the services of the SPI and the CC2420's RXFIFO. I've included the receive interrupt handler details in the CC2420 driver source code. However, before I can do the receive firmware testing, I've got to build up another host microcontroller board like the one you see in Photo 2.

Once you download the CC2420 driver source code, you'll see that is doesn't take very much to get a CC2420-based transceiver on the air. I'm really beginning to like this pointy hat stuff. With the CC2420, RF communication isn't complicated, it's embedded.

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

PROJECT FILES

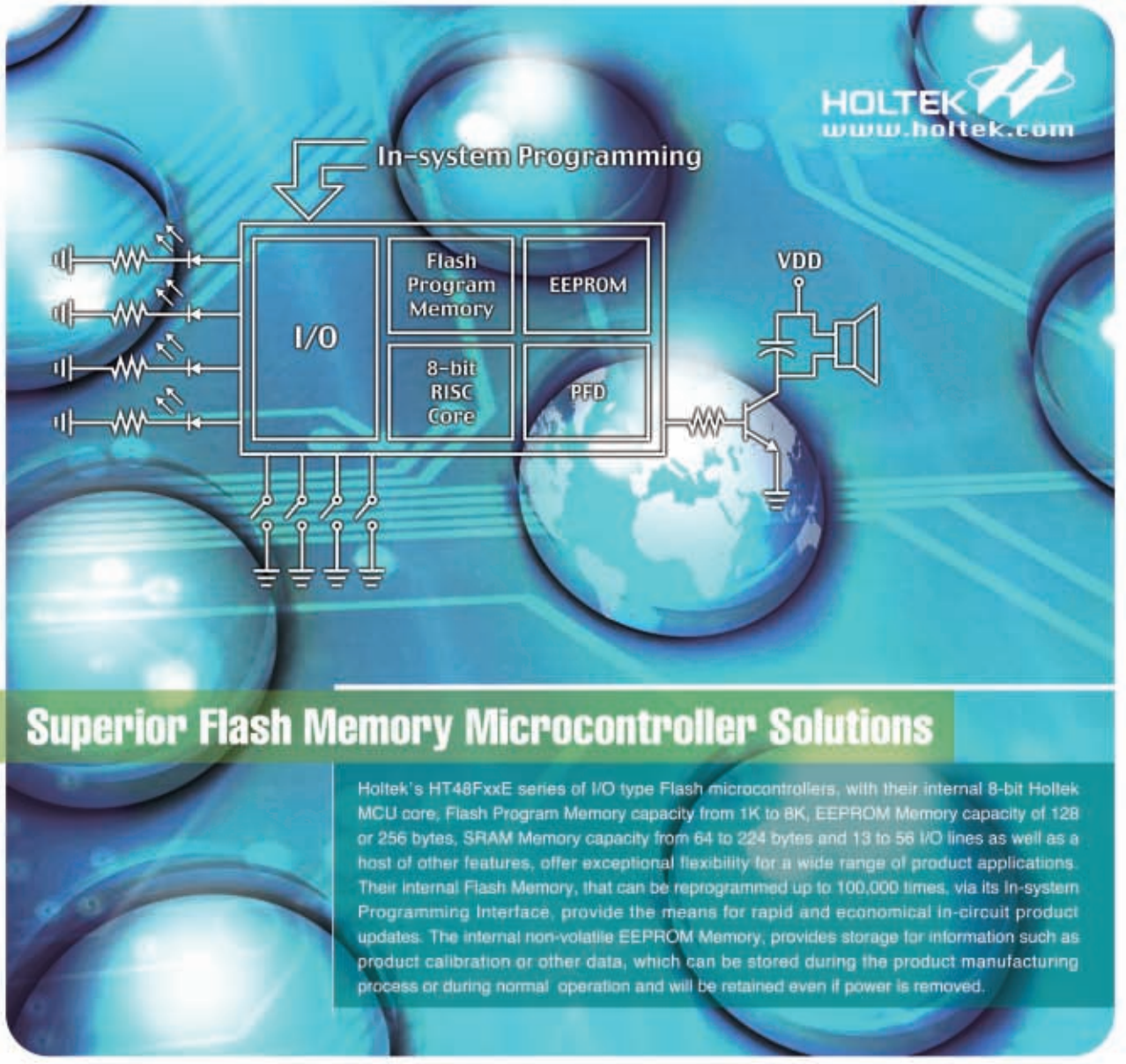
To download code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/201.

SOURCES

PICC-18 C Compiler
HI-TECH Software
www.htsoft.com

PIC18LF4620 and PICDEM Z Development kit
Microchip Technology, Inc.
www.microchip.com

CC2420 Transceiver
Texas Instruments, Inc.
www.ti.com



Superior Flash Memory Microcontroller Solutions

Holtek's HT48FxxE series of I/O type Flash microcontrollers, with their internal 8-bit Holtek MCU core, Flash Program Memory capacity from 1K to 8K, EEPROM Memory capacity of 128 or 256 bytes, SRAM Memory capacity from 64 to 224 bytes and 13 to 56 I/O lines as well as a host of other features, offer exceptional flexibility for a wide range of product applications. Their internal Flash Memory, that can be reprogrammed up to 100,000 times, via its In-system Programming Interface, provide the means for rapid and economical in-circuit product updates. The internal non-volatile EEPROM Memory, provides storage for information such as product calibration or other data, which can be stored during the product manufacturing process or during normal operation and will be retained even if power is removed.

- ◆Wide Operating Voltage: 2.2V~5.5V
- ◆Excellent Noise Immunity and ESD Protection
- ◆In-System Programming Interface
- ◆Industrial Specifications: -40°~+85° C
- ◆Flash Memory Reprogammable up to 100,000 times
- ◆Data Retention > 10 Years

Part No.	VDD	Clock	Program	RAM	EEPROM	I/O	Timer	PFD	Package
HT48F06E	2.2~5.5V	400K~8MHz	1K x 14	64x8	128x8	13	8-bit x1	V	18DIP/SOP 20SSOP
HT48F10E	2.2~5.5V	400K~8MHz	1K x 14	64x8	128x8	19	8-bit x1	V	24SKDIP/SOP
HT48F30E	2.2~5.5V	400K~8MHz	2K x 14	96x8	128x8	23	8-bit x1	V	24SKDIP/SOP 28SKDIP/SOP
HT48F50E	2.2~5.5V	400K~8MHz	4K x 15	160x8	256x8	33	8-bit x1, 16-bit x1	V	28SKDIP/SOP 48SSOP
HT48F70E	2.2~5.5V	400K~8MHz	8K x 16	224x8	256x8	56	16-bit x2	V	48SSOP 64QFP

HOLTEK SEMICONDUCTOR INC.

Holmate Semiconductor, Inc.
46729 Fremont Boulevard,Fremont, CA 94538
Tel: (510) 252-9880 Fax: (510) 252-9885
www.holmate.com



by Ed Nisley

Battery Capacity

Charge

Occasionally, paying attention to datasheet footnotes can make the difference between a successful charger and a fire hazard. This month, Ed takes a look at why NiCD and NiMH automatic battery chargers occasionally goof up.

The battery discharge characteristics you saw in my February column correspond directly to gadget performance: higher battery capacity means longer run time, higher current means more torque, and higher voltage means lower current for a given power, unless you're using throw-away alkalines. However, recharging those batteries poses an entirely different set of problems that you must solve properly to get maximum battery life at the lowest overall cost.

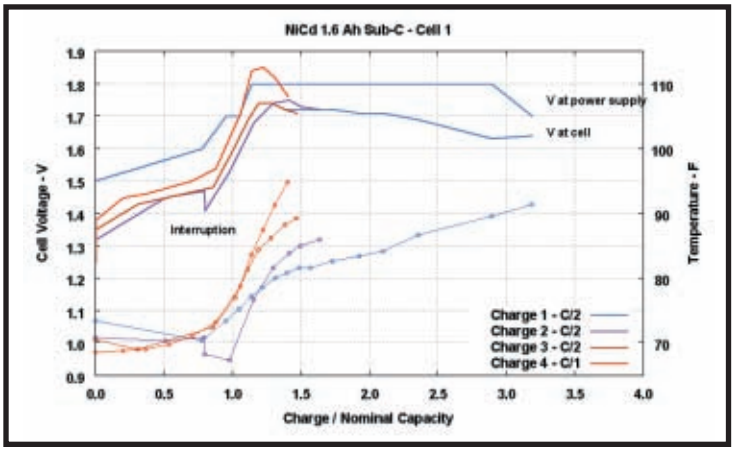


Figure 1—*NiCd cells have two full-charge indications: voltage drop and temperature rise. This cell's first charge took such a long time that it would probably fool an automatic charger.*

In this column I'll explore how NiCD and NiMH batteries behave during charging, with particular emphasis on misbehaviors that can fool automatic chargers. In most cases chargers work fine, but you'll see why paying attention to datasheet footnotes can be the difference between a successful charger and a fire hazard.

CHARGING BASICS

NiCd and NiMH cells produce electrical current from a chemical reaction that liberates electrons. Unlike nonrechargeable primary cells, a reverse electrical current reverses that reaction by supplying electrons and (more or less) restoring the cell to its original condition. Determining exactly when that reaction reaches its endpoint during charging and discharging directly affects the cell's efficiency and longevity.

Because cells produce a reasonably constant voltage during discharge, a battery's voltage does not correlate well with its remaining charge. It does behave somewhat more predictably during charging, rising monotonically as the cell's chemistry recovers. As we'll see, though, a battery's terminal voltage need not correlate with what's going on within individual cells.

As charging continues, the voltage reaches a peak and then decreases slightly after a full charge for both NiCd and NiMH cells. Depending on temperature and the charging current, which must be over about $C/2$, the decrease is about 10 to 20 mV/cell for NiCd and half that for NiMH. That's less than a 1% variation, so your measurements must have sufficient resolution to unambiguously detect the peak.

The cell temperature increases during charging and rises dramatically at full charge, at least for currents over $C/4$ in an ordinary shirtsleeve environment. Your charger must cease charging when it detects cell-damaging temperatures, a difficult

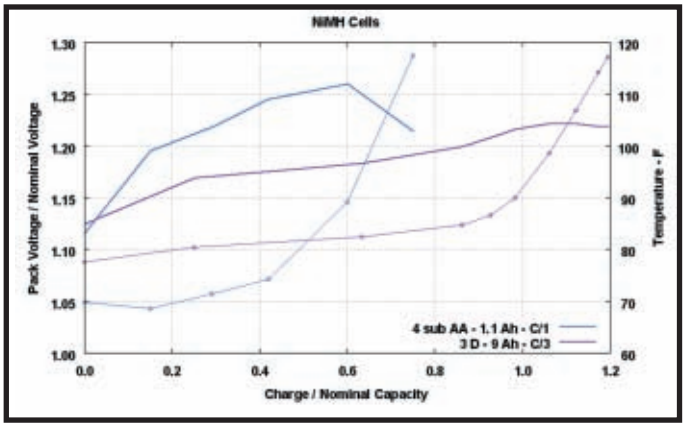


Figure 2—*NiMH cells display a much smaller voltage drop and a much faster temperature increase at full charge. You can see why NiCd chargers are not compatible with NiMH cells.*

task in extreme environments.

All of these effects are, as you'd expect, directly related to chemical conditions inside the cell. The charging reaction that absorbs external electrons is endothermic: it absorbs heat energy and the cell temperature would drop in the absence of resistive losses. That reaction ceases as the cell reaches full charge, the cell stops absorbing heat, and the temperature begins rising. The cell voltage has a negative temperature coefficient, so it decreases as the cell heats up.

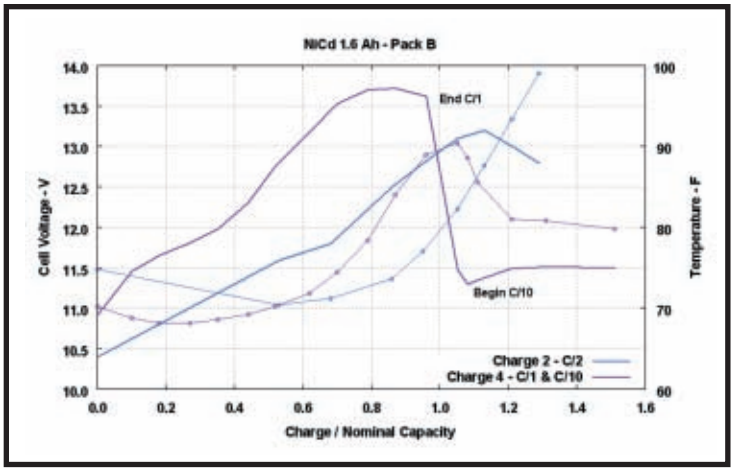


Figure 3—*Charging an eight-cell NiCd pack shows how the pack's external temperature lags its state of charge. For best results, the thermal sensor must be in close contact with the cells.*

Unfortunately, because the voltage depends on temperature, you can't use absolute temperature or voltage limits to determine the charge state. What's important is the temperature increase over the ambient and the relative voltage change from the peak. Of course, you should take ambient changes into account, but nearly all chargers assume that room temperature remains constant.

For the purposes of this column, I'll be pushing cells well past full charge to show what happens. Don't try this with cells you care about!

NEAR-DEATH EXPERIENCE

In my last column I showed the last gasp of a 10-cell NiCd battery. The pack's integrated charger was unable to fully recharge the weakest cell, which suffered reverse charging, while the rest of the pack discharged. That cell wasn't permanently damaged, most likely because I caught the problem and didn't further abuse it. I extracted the two weak cells for further testing.

The blue traces in Figure 1 show the weakest cell's first recharge. I didn't expect what was about to happen, so I began recording the cell voltage directly from the power supply's meter and only later used a DVM for higher resolution. The top two traces don't overlap because the supply's meter reported voltage drops in the wiring, while the DVM had a Kelvin connection at the cell.

That cell reached a peak voltage just over 1.7 V and simply hung there. The power supply meter can't resolve the expected 10- to 20-mV decrease, but I was looking for a temperature increase that didn't happen.

The temperature (shown in the lower blue trace, with dots) did increase, but not as dramatically as I expected. Eventually the cell voltage declined and the temperature began increasing, so I concluded the cell was as charged as it was going to get and proceeded to test it. As you saw last time, it recovered completely.

The other traces show that it behaved normally during successive charges, with 20-mV voltage decreases that correspond with the abrupt temperature increases at full charge. The peak on the red curve shows the effect of charging at $C/1$; you can see the need for temperature monitoring and a high-temp cutout.

Note that full charge occurs after the cell has absorbed about 120% of its rated capacity, starting from a completely discharged state. For this 1.6-A·h cell, that's 1.9 A·h, about 2.4 h at the $C/2$ rate. I terminated the first charge after nearly 4 h, far longer than any automatic charger would allow.

THE HEAT GOES ON

Nickel-metal hydride cells have a lower voltage temperature coefficient than NiCd cells. This is a Good Thing because the output voltage varies less with temperature, but a Bad Thing because the end-of-charge voltage decrease due to cell heating is much smaller. In fact, some sources say that you simply cannot (or should not) depend on the voltage decrease to terminate the charge. Instead, you must measure the cell temperature relative to ambient and terminate at a specific increase.

The blue traces in Figure 2 show a four-cell 1.1-A·h NiMH battery pack charged at $C/1$. The final two data points show that the temperature rose nearly 30°F in 10 min. as the pack passed through full charge, while the voltage dropped 220 mV, about 50 mV/cell.

Cell temperatures exceeding 120°F or 50°C reduce the battery's useful lifetime, so this is as hot as it should ever get. However, because I taped the thermocouple to the outside of the

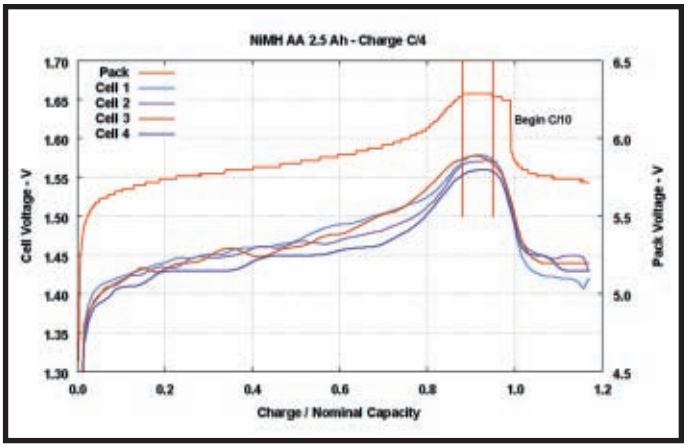


Figure 4—*Well-matched cells charge nicely together and the pack voltage represents the individual cell charge state reasonably well. Because my four-channel data logger has 20-mV resolution, I eliminated the jitter in the single-cell traces with Gnuplot's Bezier smoothing.*

pack, the internal temperature was certainly too high!

A charger designed to terminate at a drop of 10 mV/cell would have caught this pack before the temperature rose quite so high. A charger that depends only on time would severely damage this pack, because it started from a half-charged state.

The purple traces in Figure 2 show why voltage monitoring generally doesn't work for NiMH cells, even though these cells seemed to have a nice voltage drop.

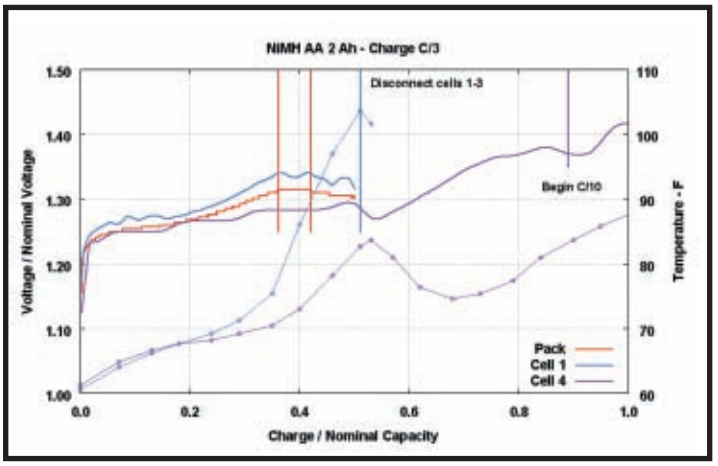


Figure 5—*I discharged Cell 4 more than the others in this pack to show the eventual effect of automatic under-charging. Terminating the charge based on pack voltage or temperature will always under-charge that cell, further weakening it. The thermocouples indicate individual cell temperatures.*

I charged three hulking 9-A·h NiMH D cells in series at 3 A, a $C/3$ rate that is the most my constant-current charger can manage. The overall voltage dropped 10 mV, only 3 mV/cell, while the temperature increased through the same 30°F. In this case, the thermocouple read the actual cell temperature.

Charging the cells at $C/2$ would increase the voltage drop, but it's obvious that the drop depends on factors that aren't completely under your control. Your charger must detect the temperature rise and terminate charging soon enough to prevent damage, while not terminating before delivering enough charging current.

Figure 3 gives a better picture of the relation between charging current, voltage drop, and temperature increase, using the eight best cells of that 10-cell NiCd pack in their original waterproof plastic container. I taped the thermocouple to the outside of the pack, where it was insulated from the cells by both an internal air gap and a few millimeters of plastic.

The $C/2$ charge, shown in blue, heats the pack slowly enough that the temperature passes through 90°F just as the voltage drops. The $C/1$ charge, however, is well past full before the heat reaches the thermocouple: it's heating at about twice the rate, so the external temperature lags the charge state by a greater margin. The temperature will rise even faster under fault condi-

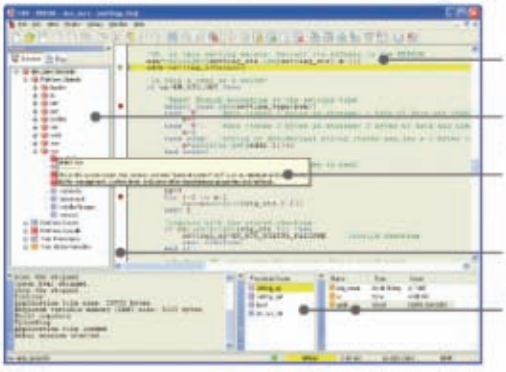
Tibbo TECHNOLOGY

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



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

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



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

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

tions, which emphasizes the need for much closer thermal contact than you see here.

PACK MATCHING

Low-voltage circuitry allows many low-power devices to run from a single alkaline or NiMH cell. High-power devices generally require several cells in series in a pack, but, inevitably, one cell must be weaker than the others. There's also one cell that's stronger than the others, but that's never a problem!

Figure 4 shows a quartet of new 2.5-A·h NiMH AA cells charging at *C*/4. The overall pack voltage (in red, right y-axis) accurately indicates the overall charge state of all four cells (various colors, left y-axis). I didn't record the temperatures during this cycle, but the *C*/4 rate just barely warmed them up by the time I reduced the current to *C*/10.

I used a four-channel, ±10-V data logger with 10-bit (20-mV) resolution and subtracted to find the differential voltage across each cell. That means the traces have up to 40-mV jitter due to sampling error alone, so I applied Gnuplot's Bezier smoothing to those four traces. The unsmoothed pack voltage has a larger scale that suppresses the small variations.

The two vertical red lines indicate times when an automatic charger might reasonably stop charging based on voltage readings. The first marks the start of the upper voltage plateau, after the final steep increase. The second marks the voltage drop after the peak.

In retrospect, those times are easy to identify while looking at the graph. An automatic charger must interpret charging's signs and portents in real-time running without any look ahead, which poses a significant challenge.

The plateau in Figure 4 lasts about 7% of nominal capacity: 17 min. at *C*/4, 8 min. at *C*/2, and just 4 min. at *C* for these cells. A typical fast charger samples the battery voltage every minute or two and, when the difference between two successive samples indicates a plateau or a decline, it terminates the charge. If all the cells have reached full charge at that time, the charger has done its job.

The raw data behind Figure 4 shows that all four cells have, indeed, reached full charge when the pack's voltage reached the plateau. Figure 5 in the February column showed that's not always the case, though, with one cell lagging far behind the others.

This month's Figure 5 shows what happens inside a deliberately sabotaged four-cell pack undergoing a moderate charge. I drained the pack by a quarter of its nominal capacity, discharged Cell 4 almost completely, and then charged the pack at *C*/3 while monitoring all four-cell voltages and the temperatures for Cells 1 and 4.

As expected, after charging for 35% of capacity, Cells 1 through 3 are fully charged, as shown by the flattening pack voltage (red trace) and Cell 1's rapidly increasing temperature (blue trace). I then disconnected the fully charged cells and continued charging Cell 4 at the same rate. All that fiddling caused the blips in Cell 4's purple trace.

Also as expected, Cell 4 required twice as much energy to become fully charged, at which point I switched to a *C*/10 trickle. Notice that the fully charged voltage for this cell is much higher than the others, which indicates that it's not in such good condition; it had been showing signs of weakness in my earlier tests, which is why I picked it for this mistreatment.

An automatic charger that would have terminated the charge at one of the times shows the red and blue vertical lines: voltage plateau, voltage drop, or temperature rise. At that point, Cell 4 was only half-charged, so it would have gone into reverse charging during the next discharge and accumulated further damage.

As nearly as I can tell, this is what kills most packs: the weakest cell suffers consistently under- and reverse-charging, which makes it even weaker, which eventually renders the pack useless.

For example, one of my amateur radio HTs has a fixed low-voltage cutoff that works fine for its standard four-cell packs, but severely over-discharges its eight-cell "high-power" packs. Despite my best efforts, I'm the proud owner of two 9.6-V packs with one dead cell in each.

CONTACT RELEASE

I used a power supply with manually controlled voltage and current regulation to produce the data for this column. This type of supply is particularly well suited to NiCd and NiMH charging because both chemistries respond well to constant-current charging. However, I had to closely monitor the charging process to terminate it at the right moment (or a bit later!) and ensure nothing went wrong.

One pack, alas, got away from me. The power supply turns on in constant-voltage mode with the current limited to 3.0 A, after which a button press activates constant-current mode at the preset value. I set a *C*/1 rate of 650 mA, set my PDA to beep in half an hour, and went off to do other things. Some time later the utility power blinked and reset the supply to 3 A.

As you might expect, ramming 3 A into a 650-mA·h battery pack causes both quick charging and severe overheating. I caught this one in time, but I now charge the cells on a ceramic tile and run the charging supply from a UPS. Charge safely!

Ed Nisley is an EE and author in Poughkeepsie, NY. Contact him at ed.nisley@ieee.org with "Circuit Cellar" in the subject to avoid spam filters.

PROJECT FILES

To download graphs, gnuplot scripts, etc., go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/201.

RESOURCES

Battery protection, www.mpoweruk.com/protection.htm

Charging overview, www.buchmann.ca/article18-page1.asp

GnuPlot, http://gnuplot.info

Logging with Dataq DI-194, users.sdsc.edu/~hubbard/neesgrid/dataq/index.html

Radio Shack 22-182 DMM, www.home.pages.at/linux/

Generic Modbus Simulator (Part 2)

Create a Modbus Master Using Visual Basic 2005

This month, Aubrey continues with the topic of a Modbus implementation on a PC. He covers creating the user interface to realize communication with Modbus slave units.

I hope I managed to whet your appetite last month when I introduced the topic of a Modbus implementation on a PC. To date, I have covered the Modbus protocol and an approach to documenting the register set of a Modbus slave and even a family of Modbus slaves in Microsoft Excel. This month, I will show you how to use that information to create a Modbus master on a PC. The software for the Modbus master is written in Visual Basic 2005 Express, which I described in my article "Visual Basic 2005 and the Serial Port," (*Circuit Cellar* 197, 2006).

I created a generic interface to allow for access to any Modbus slave via an Excel workbook as the container of the register set in each slave. My application extracts the information stored in the Excel workbook, presents it on a user interface, formats it, and sends and receives Modbus messages. The entire application is written in Visual Basic Express.

USER INTERFACE

Despite the fact that Visual Basic 2005 Express is available for free and is very easy to use, it is an incredibly powerful programming environment.[1] Of course, nothing is perfect, so you can imagine that there are flaws, some of which I will point out. This is in addition to the shortcomings that I covered in my article on the serial port.

First, you need to do some housekeeping in order to allow Visual Basic to achieve early binding (discussed in my previous article) with Excel. In technical terms, this is called adding an automation client. In Visual Basic, open or create a project, and then click Project and Add References. Next, click the COM tab. Scroll down until you see the Microsoft Excel *x.x* object library. Select the latest one (I used 9.0) and click OK. This will be fine if you are not working with my project. If you are going to open this project, you will probably end up with an error message, so here is what you should do. First, load this project. Next, click Project Modbus and Test Properties. Then click the References tab. In the References window, select the Excel and Office Object Library entries and then click Remove. Then add the automation client as above.

Documenting a Visual Basic project is quite difficult because it is all event driven. The easy way to follow the project is to right click on a control in the user form and review the properties. Double clicking on the control mostly will take you to the associated code. (I don't intend to discuss every control since the topic is dry enough. I will point out only matters that you may find interesting.)

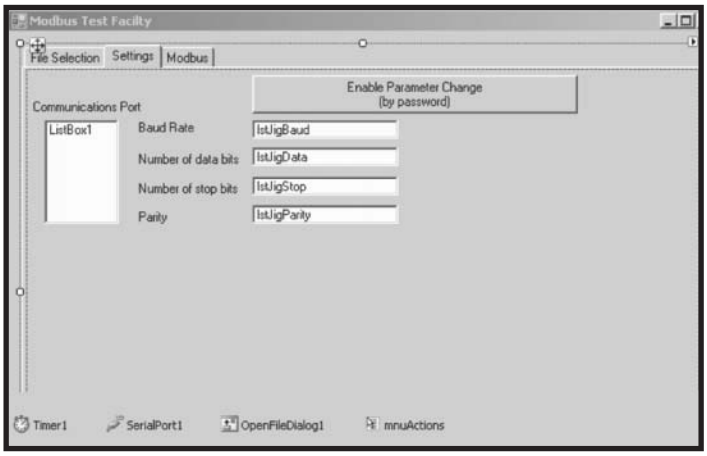


Photo 1— *The Settings tab shows the parameters that can be modified on the serial port when the application is run. Except for the port selection, the other boxes have to be enabled in order to change the values. This is done by clicking on the Parameter Change button and entering "password."*

SERIAL PORT PARAMETERS

The user interface is a tab control with three tabs (see Photo 1). The Settings tab is based on the serial port parameter selection and was described in my article on Visual Basic and the serial port. The major difference is that when the application loads, it checks to see if there is a file that contains the stored parameters. Listing 1 is added to the `Form1.Load` event.

The file save is initiated when you click on any of the other two tabs. That is when it leaves the Settings tab. The code in Listing 2 is placed in the `tabSettings.Leave` event.

FILE LOAD

The user needs to select an Excel worksheet, open it, and access the information that it contains. This is initiated from the File Selection tab in Photo 2.

In order to use the generic access to files, you must have the `OpenFileDialog` control that you dragged from the toolbox and dropped on the user form.

Clicking on the Load File button triggers the code in Listing 3. You may have noted that the file name in Listing 2 is also stored with the serial port parameters so the last accessed file name is restored when the application is rerun.

LOADING FROM EXCEL

In this application, a new instance of the Excel file is opened, even if an instance is already open. It does not pay attention to any subsequent changes to the file. Although it is open, it is invisible, and you have no visible knowledge that the file or application is running.

In Listing 4, the worksheet names, which we decided would be the product names, are loaded into the `lstWorksheetSelect` list box and sets the default selection to the first worksheet. The selection of this worksheet in the code, and any subsequent selection of any worksheet, will execute the code in Listing 5.

Let me pause for a moment and return to a few details about Excel, or rather Visual Basic within Excel. Accessing a cell from Visual Basic code can be done in two ways. You can use the range method that involves accessing a string (e.g., Range ("B5")). There is a second technique using the cells method, where the two parameters refer to the row and column. Cells (5,2) refer to cell B5 as well. It stands to reason that programmatically the cells approach would be easier. I have no idea if it was Visual Basic 2005, Excel 2002, or that I didn't quite create the right object (although I suspect the latter), but I could not get the cells method to be exposed. As a result, all the arithmetic manipulations that had to be done to access a particular cell had to undergo an additional transformation to a string. The irony is that in the Visual Basic Excel help file (`GetObject` Function), Microsoft explicitly frowns upon doing it this way. But for me, it was the only way that worked.

Listing 1—*This code is added to load serial port settings from a file that is stored in the same directory as the application. If it fails (i.e., the file does not exist), it forces the user to the Setup tab to review the default settings.*

```
sPath = Application.StartupPath & "\mod.cfg"
'start in the same directory as the application
Try
    sLines = System.IO.File.ReadAllLines(sPath)
    sJigPort = sLines(0)
    sJigBaud = sLines(1)
    sJigBit = sLines(2)
    sJigStop = sLines(3)
    sJigParity = sLines(4)
    Target.sFileString = sLines(5)
    ListBox1.Text = sJigPort
    lstJigParity.Text = sJigParity
    lstJigData.Text = sJigBit
    lstJigBaud.Text = sJigBaud
    GenerateJustFileName()
    Me.SelfTest.SelectedTab = tabFileSelection
    OpenPorts()
 Catch ex As Exception
    'failed to open file
    MsgBox(ex.Message & vbCrLf & "Please set up parameters", _
    MsgBoxStyle.Exclamation, "First Time Setup")
    Me.SelfTest.SelectedTab = tabSettings
    'force to settings tab
End Try
```

Loading the profile names to the `lstProfileItems` list box can be seen in Listing 6. It will give you a taste of the string manipulation that I am talking about.

The listing for `LoadAddresses FromExcel` is rather long and boring. If you have managed to read to this point, I am sure you have opened the Visual Basic project where you can see it in all its glory. It only differs from Listing 6 in that it reads two columns and merges them to create a line including the address and the location name. It also looks for blank addresses. If it finds them, it inserts a line of dashes to indicate the compression. This list box is the heart of all the actions and information in it is used in a number of procedures and in the interchanges with Excel. I decided to centralize the information by creating a class (see Listing 7). It is instantiated as an object named `User`. Two look-up tables allow for a speedy cross-reference between Excel and the application. They are stored in

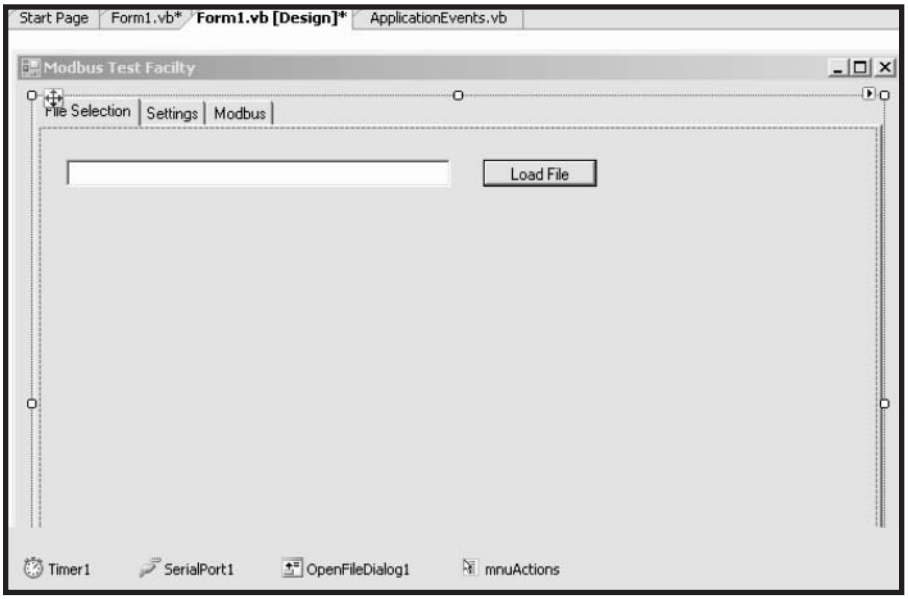


Photo 2—*To access an Excel file, click on the Load File button. Using the standard OpenFileDialog, locate the desired file. Note the COM and Dialog controls that have been placed on the form.*

two arrays, `iExcelToIndex()` and `iIndexToExcel()`, which cross-reference the line index in the `lstAddrEnvir` list box to the rows in the Excel worksheet. I will discuss the other members of the class in due time.

THE APPLICATION USER INTERFACE

In order to access a Modbus location once the Modbus device has been selected (by selecting the worksheet), you need to decide several things. You must decide whether you want a single or multiple location access, the start address in either event, and the number of locations in the latter case. Two radio buttons allow for single or multiple selections. The rest of the choices are handled in the list box with the location names and addresses.

Despite similarity in appearance to Windows controls, I found that the list box control lacks the methods to do certain things that appear to be standard in Windows. My original intention was to click on an address and then Shift-click on the end address so all the lines would be selected. It may be possible to do, but you will see that it and other desirable features (at least to my way of thinking) are not built in. For instance, I am still not sure how to change the font and color of a sub string within an entry. Modifying a line requires you to read back the line, delete it, and reinsert the modified line. Changing the color of individual lines requires you to write a handler that runs each time the list box is redrawn. Enough of the griping, let's see how I tackled this.

I decided that right-clicking in the `lstAddrEnvir` list box and choosing from a drop-down menu would realize most of the control I wanted. I first needed to drag a `ContextMenuStrip` control from the toolbox, drop it on the form , and name it. The next step was to bind the menu control to the associated list box. In the properties for the `lstAddrEnvir` list box, I inserted the name of the menu control in the `ContextMenuStrip` property. I then added the following options to the menu: Set Start Address, Set End Address, Clear All Addresses, Clear All Data, and Set Data. Each one of these leads to associated code. The Set

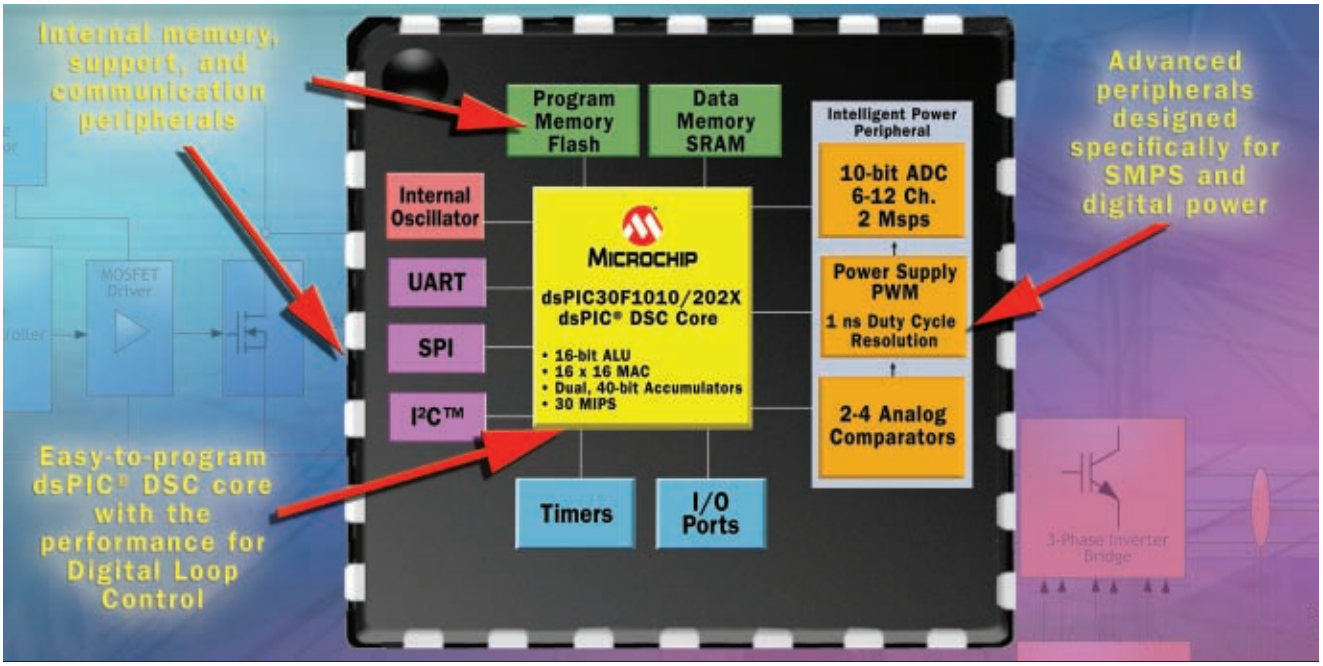
Listing 2—*Write serial port parameters to a file when leaving the Settings tab. Note that in order to change the parameters, you need to enable the associated boxes using the `EnableParameters` procedure, which requires a password: "password." Not very original! Every time the Settings tab is selected, the boxes will be initially disabled.*

```
Private Sub TabPage5_Leave(ByVal sender As Object, _
ByVal e As System.EventArgs) Handles tabSettings.Leave
    'updating the parameters to the new settings
    sJigPort = ListBox1.Text
    sJigBaud = lstJigBaud.Text
    sJigBit = lstJigData.Text
    sJigStop = lstJigStop.Text
    sJigParity = lstJigParity.Text
    WriteConfigurationFile()
    'write to file
    OpenPorts()
End Sub
Public Sub WriteConfigurationFile()
    Dim sPath As String
    Dim sLines(NUMBER_OF_PARAMETERS) As String
    sPath = Application.StartupPath & "\mod.cfg"
    sLines(0) = sJigPort
    sLines(1) = sJigBaud
    sLines(2) = sJigBit
    sLines(3) = sJigStop
    sLines(4) = sJigParity
    sLines(5) = Target.sFileString
    System.IO.File.WriteAllLines(sPath, sLines)
End Sub
Private Sub TabPage5_Enter(ByVal sender As Object, _
ByVal e As System.EventArgs) Handles tabSettings.Enter DisableParameters()
    ClosePorts()
End Sub
Sub DisableParameters()
    lstJigBaud.Enabled = False
    lstJigData.Enabled = False
    lstJigParity.Enabled = False
    lstJigStop.Enabled = False
End Sub
Sub EnableParameters()
    lstJigBaud.Enabled = True
    lstJigData.Enabled = True
    lstJigParity.Enabled = True
    lstJigStop.Enabled = True
End Sub
```

Listing 3—*Find a file using the generic `OpenFile` procedure. Only the file name (without path) is displayed in the list box.*

```
Private Sub Button1_Click(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles btnLoadFile.Click
  OpenFileDialog1.InitialDirectory = Application.StartupPath
  OpenFileDialog1.Title = "Select an Excel file..."
  OpenFileDialog1.Filter = "Excel files | *.xls"
  OpenFileDialog1.FilterIndex = 1
  OpenFileDialog1.FileName = ""
  If OpenFileDialog1.ShowDialog() <> Windows.Forms.DialogResult.Cancel Then
  Target.sFileString = OpenFileDialog1.FileName
  'create file name without path for display
  GenerateJustFileName()
  Else
  txtLoadedFile.Text = ""
  End If
End Sub
Public Sub GenerateJustFileName()
  Dim iI As Integer
  Dim iJ As Integer
  iI = InStrRev(Target.sFileString, "\")
  iJ = Microsoft.VisualBasic.Len(Target.sFileString)
  txtLoadedFile.Text = _
  Microsoft.VisualBasic.Right(Target.sFileString, iJ - iI)
End Sub
```

Turn Digital Power into Real Power



Looking to bring the benefits of digital power conversion – including higher power density, lower cost features and accelerated innovation – to your product?

Microchip's dsPIC® Digital Signal Controllers (DSCs) for Switch Mode Power Supplies (SMPS) and digital power conversion are designed for applications in AC/DC power supplies, power factor correction, DC/DC converters, uninterruptible power supplies, power inverters, digital lighting and advanced battery chargers.

GOING DIGITAL THE EASY WAY!

1. Educate yourself
Learn the technical aspects of digital power conversion and SMPS design with hands-on training and ***FREE*** web seminars from Microchip.

2. Accelerate your development



Speed up your design time with tools targeting digital power such as the **dsPICDEM™ Buck Development Board**.

3. Win stuff in our Digital Power Contest
Visit our web site and get coupons for discounted development tools when you ***REGISTER TO WIN*** a **dsPIC DSC SMPS Development System**. We are also giving away an Apple® iPod® mobile digital media player to one lucky winner! Visit our web site for complete contest information.

Purchase and program your 16-bit dsPIC® DSCs and related development tools at....

microchip DIRECT
www.microchipdirect.com

Device	Pins	Flash (KB)	10-bit, 2 MSPS (CH)	High Speed PWM (CH)	Analog Compar-ators
dsPIC30F1010	28	6	6	4	2
dsPIC30F2020	28	12	8	8	4
dsPIC30F2023	44	12	12	8	4

Visit our web site for more information about additional 16-bit devices supporting power conversion!

MICROCHIP
www.microchip.com/SMPS

Microcontrollers • Digital Signal Controllers • Analog • Serial EEPROMs

The Microchip name and logo, the Microchip logo and dsPIC are registered trademarks of Microchip Technology Incorporated in the USA and in other countries. dsPICDEM is a trademark of Microchip Technology Incorporated in the U.S.A. and other countries.
© 2007 Microchip Technology Incorporated. All rights reserved.

Start Address option is self explanatory, setting the `User.iStartAddressIndex` and `User.bStartSetUp` variables. The `Set End Address` acts similarly, but it is only needed for multiple register accesses. Obviously, the Clear All Addresses resets the above settings. There are, of course, checks to make sure that the start address precedes the end address, that the grouping is only of like registers, and so on.

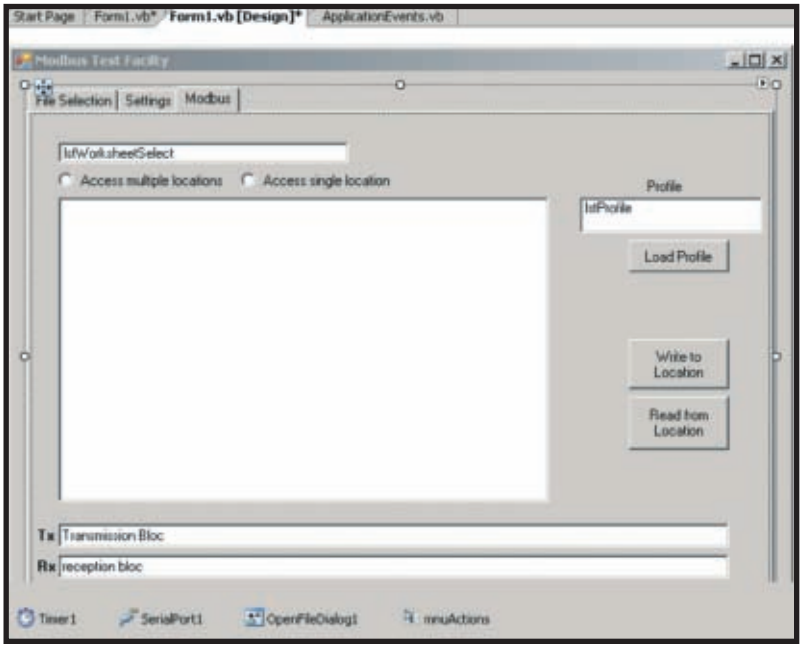


Photo 3—*Take a look at the construction details of the tab.*

The normal color of the text in the user box is orange, which shows up well against the dark blue of the reverse video selection indicator. The color of the text of any line selected between the start and end address is turned to red. I also chose the Courier font to ensure equal spacing of any character so that the entries line up. To achieve the color change, I needed to change the DrawMode property of the `lstAddrEnvir` list box to `OwnerDrawFixed`. Look at the `lstAddrEnvir_DrawItem` procedure to see the code that is needed.

Aside from loading profiles, as we saw earlier, it is also possible to insert values to be written to a register. The Set Data brings up an Inbox to enter numerical data. When this is entered, the selected line must be updated by reading, deleting, and inserting, as I complained about earlier. The Clear All Data goes through all the lines clearing all data, whether it was set by the Load Profiles, Set Data, or even data read back from a Modbus slave. It too must go through the same update procedure.

MODBUS

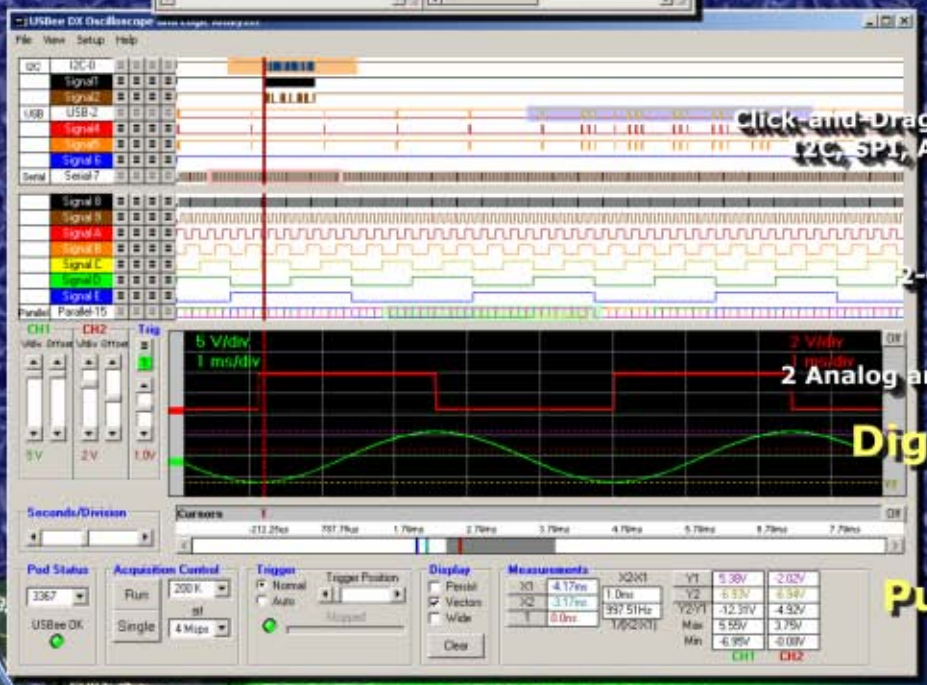
I have now come full circle. Let's take a look at the interactions over

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
On the web at www.jkmicro.com
JK microsystems

Wireless Control Application Kit
900 MHz and 2.4 GHz RF Control
• Low power LP3500 Single-Board Computer
• Two MaxStream® license-free RF Modules, FCC approved
• Software libraries and samples
• ModBus and PPP support for serving web pages
Complete Application Kit
$599 2.4 GHz Version
$499 900 MHz Version
RABBIT Semiconductor
Order Online At rabbitappkits.com
07070

USBee DX





Oscilloscope/Logic Analyzer

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

Bus Decoding

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

Digital Voltmeter

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

Data Logger

2 Analog and 16 Digital Signals Plus Timestamp

Digital Signal Generator

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

Pulse Width Modulator

16 User Controlled PWM Channels

Frequency Counter

16 Channel Counter to 24MHz

Frequency Generator

Generates Sets of Common Frequencies

I2C Controller

Control I2C Devices Using Simple Text Scripts

Remote Controller

Easily Control Your Hardware Using Your PC

Pulse Counter

16 Channel Pulse Counter With Gating Control

plus the USBee Toolbuilder Source code and Library

Create Your Own Applications to Control the USBee DX



Data Extractors

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

Continuous Real-Time Embedded Bus Data Streaming

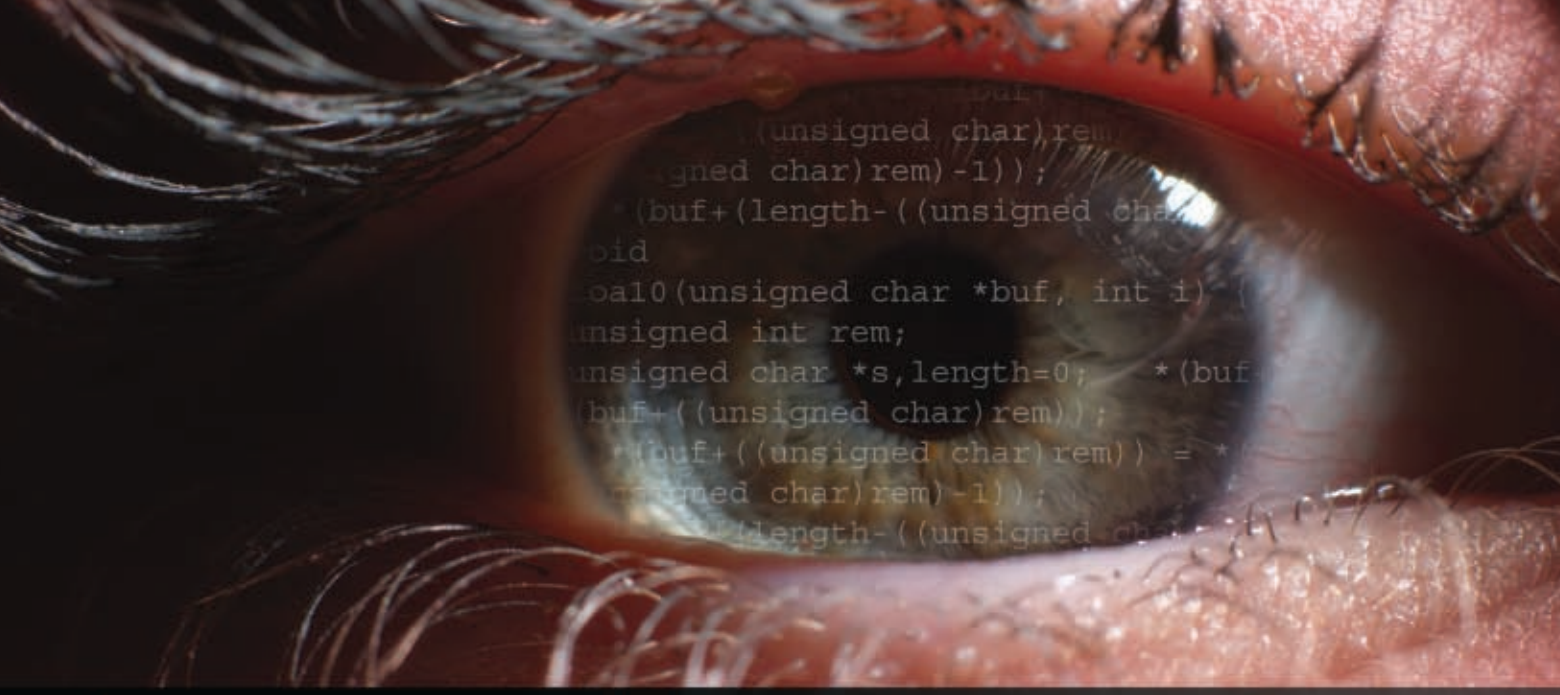
Store Bus Data to Disk or Send to Your Custom Application

Capture and Process Entire Test Sequences

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



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



focus more on your goals and less on your code

NEW!!

PICC-18™ PRO with OMNISCIENT CODE GENERATION™

PICC-18™ PRO ANSI C COMPILER*, based on the reliable PICC-18™ Standard compiler, the PICC-18™ PRO comes with ...

OMNISCIENT CODE GENERATION™
Extracts information from multiple source files at the one time, allowing more intelligent state-of-the-art code generation that ...

- Optimizes the size of each pointer variable in your code based on its usage.
- Eliminates the need for many non-standard C qualifiers and compiler options.
- Produces more optimal interrupt context switching code.
- Customizes the functionality of the included printf library function.
- Is simple to use.

** Can run on multiple platforms including **Windows, Linux and Mac OSX.***

PICC-18™ PRO is packaged with...

HI-TECH PRIORITY ACCESS™ (at no extra cost): HI-TECH Software's 12 month **web access to updates** and **priority technical support** (both phone and email) from arguably the industry's best team of C compiler engineers.

HI-TECH SATISFACTION GUARANTEE
HI-TECH Software offers a 30 day money back guarantee. If for any reason you are not completely satisfied with our product, simply contact us and we'll refund the purchase price.

MICROCHIP MASTERs

HI-TECH Software presenting August 8-11. For more details visit www.htsoft.com.

Introductory Offer: Save $300. Find out more or download a fully functional demo at http://www.htsoft.com/products/picc18procompiler.php

HI-TECH SOFTWARE

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

HI-TECH Software proudly supports the Microchip brand with industrial-strength software development tools and C compilers.

Recognized Microchip Third-Party Tool Provider

PICC-18™ is licensed exclusively to HI-TECH Software by Microchip Technology Inc.

the serial port. Clicking on Write To Location starts a process that fetches the function codes to be used from the Excel worksheet (from columns G, H, and I). It also makes sure that the locations can be modified by testing that the function code is applicable. The data to be written to the slave is extracted from the `lstAddrEnvir` display. If there is no entry, the data value is set to zero. For digital (coil) outputs, any nonzero value will be sent as a set command. Depending on the decision to access single or multiple locations, different messages are created. I daresay that it is possible to create a single function that handles all the write functions and the read functions, but I decided to be modern for once in my life and I didn't care about the amount of memory that it would chew up. Please forgive my lack of programming elegance.

Creating the message to send as the master is trivial. Simply fetch the required information and plug it into the frame format described in Part 1 of this series of articles (*Circuit Cellar* 200, 2007). The only pause is to pack (in `boolControlMultipleBits`) and unpack (in `bitReadMultipleBits`) the bit format into bytes and to generate the CRC (done in the function `cGenerateChecksum`). As an aside, you may have already noted a number of modifications and additions to the Visual Basic language that resemble C. At long last, it is possible to shift/rotate data instead of multiplying or dividing by two. The notation is identical in both languages using the "<< " and ">>" symbols.

Once the actual message has been created, it is written to the Tx text box at the bottom of the form before it is transmitted from the serial port. Once the frame has been sent, the process starts the `bytReceiveModbus` function that waits for the return message. Modbus uses variable-length messages, so the software has to dynamically extract the message length from the frame as it is received. The function returns an array where the first byte contains the status, as you can see in the comments in Listing 8. Although I could have put the CRC check in this function, I chose not to because I wanted the whole stream to be returned so that it could be reported to the Rx text box on the user interface. My concept was that this should be updated outside of the byte-reception function.

Once the software process returns from the byte-receive function, the datastream is written to the Rx box. Next, the process checks for an error return in the first byte of the array and then it checks to see if the CRC of the message is consistent with the expected value. Several other checks may also be done on the byte stream, depending on the function being performed. The analysis of all these checks is presented as Label 7 at the bottom of the form. These could include messages, such as "Response received correctly" or "CRC error detected," to help with the understanding of the Modbus interaction since this application is intended to be a development tool. If there is returned data, it is then updated in the associated data field in the `lstAddrEnvir` list box overwriting any data that may have been there

Listing 4—*When the user clicks on the Modbus tab, the file selected in the OpenFile dialog is opened as an Excel application. When the Modbus tab is deselected by selecting another tab, the Excel application is terminated. Note how to create the different objects to access Excel.*

```
Private Sub tabModbus_Enter(ByVal sender As Object, _
ByVal e As System.EventArgs) Handles tabModbus.Enter
 StartExcelApplication()
 ClearModbusMessages()
End Sub
Public Sub StartExcelApplication()
 Dim iI As Integer
 Dim iNumberOfSheets As Integer
 Dim sSheetName(100) As String
 Target.oXL = CType(CreateObject("Excel.Application"), Excel.Application)
 Target.oWB = Target.oXL.Workbooks.Open(Target.sFileString, , True)
 For iI = 1 To 100
 'finding the number iof sheets and their names
 Try
       Target.oSheet = Target.oWB.Worksheets(iI)
 sSheetName(iI - 1) = Target.oSheet.Name
  Catch ex As Exception
       Exit For
 End Try
 Next
 iNumberOfSheets = iI - 1
 'using this information to create options in a list box
 For iI = 0 To iNumberOfSheets - 1
       lstWorksheetSelect.Items.Add(sSheetName(iI))
  Next
 'select the first one
 lstWorksheetSelect.Text = sSheetName(0)
 Target.oSheet = Target.oWB.Worksheets(1)
 'index of ssheetname and worksheets are offset by 1
 Target.oSheet.Activate()
 ClearUserSettings()
End Sub
Public Sub EndExcelApplication()
 Target.oXL.Workbooks.Close()
 Target.oXL.Workbooks.Application.Quit()
 Target.oRng = Nothing
 Target.oSheet = Nothing
 Target.oWB = Nothing
 Target.oXL = Nothing
 ClearUserSettings()
End Sub
Private Sub tabModbus_Leave(ByVal sender As Object, _
   ByVal e As System.EventArgs) Handles tabModbus.Leave
       EndExcelApplication()
End Sub
```

from reads or preset data.

PUBLISHING THE PROJECT

Creating an executable to run outside of the Visual Basic environment is termed "publishing," and it is simple in the extreme, both to generate and to install. First, you need to set the project properties, especially the assembly information under the Application tab. This sets up the revision and copyright details. You may notice an option to set the "splash screen." Depending on your operating system and some other details, the splash (start-up) screen option may create run-time errors. It certainly did on mine (with Microsoft acknowledging the problem), so I will hold off on even trying to describe the process. You then click on Build, Publish ProjectName, and follow the prompts in the Publish wizard. Then, publish it to a CD-ROM. Create a folder and the application will be placed in the folder. Distribution of the application merely involves transferring this folder. A setup.exe file is in the folder that you can rename to any other name, with an .exe file extender.

Installation is very simple. Copy the folder to wherever you want it and double click on the setup.exe (or its new name) file. The first time you do this it will be installed to your startup menu and the program will run. In this application, you will see the error message that there is no set-up file and one must be created. All of the subdirectories are hidden within the folder. It is possible to run the application at any time after that by double clicking the self-same file or by using the entry on the Start menu. Of course, you could copy a shortcut onto the desktop.

Uninstalling is a cinch. Use the uninstall option. Go to My Computer, Control Panel, and then Add/Remove Programs.

CONVERGENCE

In this series of articles, I covered many aspects of electrical engineering: hardware and hardware interfacing through the serial port, an approach to operating Excel as an external object, the programming environment created by Visual Basic, and an external serial protocol, the Modbus. Convergence with a vengeance! I hope this series has proved informative.

Aubrey Kagan (antediluvian@sympatico.ca) is a professional engineer with a B.S.E.E. from the Technion—Israel Institute of Technology and an M.B.A. from

Listing 5—*This code is run any time the Excel worksheet selection is changed by clicking on an item in the* `lstWorksheetSelect` *box. It calls the* `LoadAddressesFromExcel` *procedure, which fetches the addresses, compresses, and displays them in the* `lstAddrEnvir` *box, which is shown in Photo 3. It also loads the profiles to the* `lstProfile` *box.*

```
Private Sub lstWorksheetSelect_SelectedIndexChanged _
(ByVal sender As System.Object, ByVal e As System.EventArgs) _
Handles lstWorksheetSelect.SelectedIndexChanged
    Dim iI As Integer
    iI = lstWorksheetSelect.SelectedIndex
    'iI = lstWorksheetSelect.SelectedItem
    Target.oSheet = Target.oWB.Worksheets(iI + 1)
    'index of ssheetname and worksheets are offset by 1
    Target.oSheet.Activate()
    LoadAddressesFromExcel()
    LoadProfileNames()
    lstAddrEnvir.Focus()    'make sure that if the scroll
    'button is the next control after setting the page,
    'the page selection does not scroll.
    radMultiple.Checked = True'set by default to multiple addresses
    ClearUserSettings()
    ClearModbusMessages()
End Sub
```

Listing 6—*Initially,* `iTemp` *gets the number of profiles from the location D5 on the active sheet. The active sheet was set in Listing 5. The procedure then accesses row 7 column by column, starting at column J. It does this by converting the ASCII representation of J to an integer* `Asc "J,"` *adds an offset to it, converts it back to ASCII, and then concatenates it with row "7" to create the variable* `sString.sString` *is then used as the parameter in the Range method.* `CStr` *casts the value returned from Excel to a string to add as an item in the list box.*

```
Private Sub LoadProfileNames()
   Dim iI As Integer
   Dim sString As String
   Dim iTemp As Integer

   'first clear the entries
   lstProfile.Items.Clear()
   iTemp = Target.oSheet.Range("D5").Value
   If iTemp <> 0 Then
   For iI = 0 To iTemp - 1 'NUMBER_OF_PROFILES - 1
        'create a string incluidng address of cell
   sString = Chr(Asc("J") + iI) & "7"  'creating string j, k, l...
   lstProfile.Items.Add(CStr(Target.oSheet.Range(sString).Value))
   Next iI
   lstProfile.SetSelected(0, True) 'by default set the first
   End If
End Sub
```

Listing 7—*The* `UserSelection` *class is instantiated as the object* `User`*, using a* `dim` *statement at the start of the* `Form1` *class (see the project itself).*

```
Public Class UserSelection
  Public bStartSetUp As Boolean
  Public bEndSetUp As Boolean
  Public iStartAddressIndex As Integer
  Public iStopAddressIndex As Integer
  Public iExcelToIndex() As Integer
  'should allow for quick lookup from one to the other
  'especially since there are lines of --- which could cause a problem
  Public iIndexToExcel() As Integer
End Class
```

the University of the Witwatersrand. He works at Emphatec, a Toronto-based design house of industrial-control interfaces and switch-mode power supplies. In addition to writing several articles for Circuit Cellar *and having ideas published in other periodicals, Aubrey wrote* Excel by Example: A Microsoft Excel Cookbook for Electronics Engineers *(Newnes, 2004).*

Listing 8—*The byte reception is modified from the original code in my first article. The number of bytes expected depends initially on the received function code (including the Modbus exception). In some cases, the number of bytes is explicitly embedded in the message and must be extracted.*

```
Function bytReceiveModbus _
(ByVal bySentAddress As Byte) As Array
    'start timer
    Dim iRx(255) As Byte
    Dim iI As Integer = 0
    Dim iNumberOfBytes = 250

    bTimeout = False
    Timer1.Interval = 1000     '1 sec delay
    Timer1.Start()

    iRx(0) = 0     'set return flag for OK
    'on return
    'if iRx(0)=0 ->OK
    'if iRx(0)=1 ->Timeout
    'if iRx(0)=2 ->Modbus error
    'if iRx(0)=3 ->Address mismatch

    While (iI < iNumberOfBytes) And (bTimeout = False)
    'so don't overflow buffer

    'only while timing out
    Application.DoEvents()
    If SerialPort1.BytesToRead > 0 Then
       iRx(iI + 1) = SerialPort1.ReadByte
       'now checking on the numebr of bytes
       If iI = 1 Then
         'this is the 2nd byte-> if bit 8 is set then fault
         'and length is 5
         If (iRx(2) And &H80) = &H80 Then
            iNumberOfBytes = 5
            iRx(0) = 2
       End If
       End If
       If (iI = 2) And ((iRx(2) And &H80) = 0) Then
          'no error, and enough bytes to get number of bytes
          Select Case iRx(2)
           Case 5, 6, 15, 16
              iNumberOfBytes = 8
           Case 1, 2, 3, 4, 12, 17, 23
              iNumberOfBytes = iRx(3) + (5)
           Case 7
              iNumberOfBytes = 5
           Case 11
              iNumberOfBytes = 8
           Case Else    '8
          End Select
       End If
       iI = iI + 1
    End If
    End While
    If bTimeout = True Then
       iRx(0) = 1
       iI = 0
    Else
       If (iRx(1) <> bySentAddress) Then
          iRx(0) = 3 'use address as fault indicator
          iI = 0
       End If
    End If
    ReDim Preserve iRx(iI)
    bytReceiveModbus = iRx
End Function
```

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/201.

REFERENCE

[1] Microsoft Corp., "The Microsoft Developer Network," 2006, http://msdn.microsoft.com/vstudio/express/vb.

RESOURCES

F. Balena, *Programming Microsoft Visual Basic 2005: The Language*, Microsoft Press, 2006.

J. Foxall, *Visual Basic 2005 in 24 Hours*, SAMS Publishing, 2006.

M. Halvorson, *Microsoft Visual Basic 2005 Step by Step*, Microsoft Press, 2006.

Microsoft Corp., "How to Automate Microsoft Excel from Visual Basic .NET," 301982, http://support.microsoft.com/kb/301982.

———, "How to Transfer Data to an Excel Workbook Using Visual Basic .NET," 306022, http://support.microsoft.com/kb/306022.

———, "How to Use MFC to Automate Excel 2000 and Excel 2002 and Obtain an Array From a Range in Excel 2000 and Excel 2002," 186122, http://support.microsoft.com/kb/186122.

Modbus-IDA, "MODBUS Over Serial Line, Specification & Implementation Guide V1.1," 2002, www.modbus.org/docs/Modbus_over_serial_line_V1.pdf.

———, "MODBUS Application Protocol Specification V1.1a," 2004, www.modbus.org/docs/Modbus_Application_Protocol_V1_1a.pdf.

———, "Modicon Modbus Protocol Reference Guide Rev J," PI-MBUS-300, 1996, www.modbus.org/docs/PI_MBUS_300.pdf.

P. Pelland, *Build a Program Now! Microsoft Visual Basic 2005 Express Edition*, Microsoft Press, 2006.



Local Interconnect Network

Instead of a controller area network (CAN), try a local interconnect network (LIN) the next time you need to control a limited area within a machine. Jeff gets you working with a low-cost LIN system.

In an attempt to steer my focus back toward an exploration of electric vehicles and their subsystems, I want to start investigating a communication medium that's becoming popular in the automotive world: the local interconnect network (LIN) bus. Although the LIN is thought of as a lower-bandwidth medium, it seems plenty adequate for my project. This column will cover uses of the LIN in distributed control. My plan involves a small cluster of LIN modules, consisting of a master and at least two slaves at this point.

LIN is used as a low-cost and low-end solution for distributed control in the automotive industry (see Figure 1). Its UART-based single-wire bus is about as simple as it gets. LIN is not meant to be a replacement for the controller area network (CAN), but it's an alternative where requirements are not as stringent. LIN's bandwidth is presently limited to approximately 20 kbps. It is often used within a specific area, such as a door assembly. While a manufacturer might use a CAN bus to connect various components of an automobile, it may make more sense to use local control within those components. For instance, within a door, a side mirror, a window system, a lock, and user controls.

Manufacturers of individual components are more likely to invest in this approach if they can be assured that their equipment will "plug and play" well with others. To ensure that this will be true, the LIN consortium was formed in 2000 to oversee the standardization of this serial low-cost communication concept, including a development environment. Original members include Audi, BMW, DaimlerChrysler, Volvo, and Volkswagen, along with communications specialist Volcano Communications Technologies and the semiconductor manufacturer Motorola.

The open-standard LIN specification package (V2.0) consists of several parts. The LIN's physical layer specification describes the physical layer, including the bit rate, clock tolerances, etc. The LIN protocol specification describes the data-link layer of the LIN. The LIN diagnostic and configuration specifications describe the service that can be layered on top of the data-link layer to provide diagnostic messages and node configuration. The LIN API specification describes the interface between the network and the application program, including the diagnostic module. The LIN configuration language specification describes the format of the LIN descriptor file (LDF). The LDF is used to configure the complete network and serve as a common interface between the original component manufacturer (OEM) and suppliers of the different network nodes. It is also used as an input to development and analysis tools. The LIN node capability language specification describes a format used to describe off-the-shelf slave nodes in a node capability file (NCF) that can be used with a plug-and-play tool to automatically create an LDF.

Prior to the release of V2.0, there was little flexibility in the slave node. Each slave was hard-coded with its choice of identity. You can imagine that with a limited number of identifiers (60), it wouldn't take long for someone to use two nodes programmed with identical identifiers. Even though a master node is in charge, there was little it could do before V2.0 about slaves with conflicting NADs or `message_ID`s.

The consortium's steering group solved many shortcomings in the previous release by designating a development workflow that not only included input from slave nodes (NCF), but also allowed conflicts to be mediated prior to assembly of the cluster (LPF). With this release, nodes must be reconfigurable (i.e., it gets its frame identifiers assigned by the master node according to the LPF). The block diagram in Figure 2 shows how the cluster design phase uses all NCFs to create the LDF. The master is created from the LDF and passes this configuration information on to all of the slaves.

START AT THE END

To see how this all works, it makes sense to start at the end of the LIN specification list. Before you take control of something, you have to know what its capabilities are. The manufacturer of any LIN-compatible node

must provide an NCF documenting the node's abilities. This file's ASCII structure makes it possible for the file to be read by technicians and LIN tools developed to help LIN managers eliminate errors (and even produce code.)

The NCF is a list of node definitions including areas for `general`, `diagnostic`, `frame`, `status_management`, and `free_text` definitions (see Listing 1). The supplier is a member of the LIN consortium. You can find its respective supplier_id on www.lin-subbus.com. Suppliers are responsible for issuing a function_id for each product manufactured under their supplier_id. When a product undergoes revision, which does not affect the existing function, the manufacturer increments the variant_id. Conformance testing is available to assure adherence to the LIN specification.

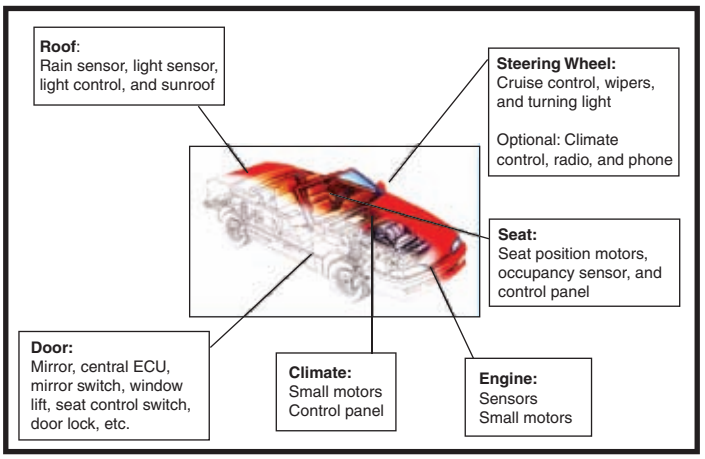


Figure 1—*This diagram shows a number of typical areas where using individual LIN buses makes sense.*

The frame definitions can have multiple entries with optional parameters for each `message_id` entry. The `publish` frame declares the information it has available to others. The cluster's master will use this frame to gather information, the slave will respond, and any node (including the master) can use the data. A `subscribe` frame describes a function that it can perform. In this case, a cluster master will use this frame to produce this information, a slave will reply (it could in fact be the master), and any node can use it. The master always initiates a frame as guided by the LDF. In most cases, only one slave (or the master itself) will respond.

LIN DESCRIPTION FILE

With all cluster nodes defined in separate NCFs, these documents become the input to the master node's LDF. The LDF is similar to the NCF. It contains sections describing the `cluster node`, `signal`, `frame`, `node_attributes`, and `schedule_table`. While most of the information is garnered from the NCFs, the LDF must reconcile conflicts and create a communication schedule.

The first area of concern is with the NAD chosen by each node. Every node within the cluster must have a unique NAD. If multiple nodes declare identical NADs in their NCF, the master must reassign a new NAD. It will do this via a diagnostic frame, using the protected identifier 0x3C (60). The master node is able to communicate with each of the slaves diagnostically, via the declared or assigned NAD.

Cluster communication is normally handled by protected identifiers, 0-0x3B (0–59). The master must assign a unique protected identifier to each `published` and `subscribed` frame of every node. Each node has already declared a `message_ID` for each of its own `published` and `subscribed` frames. The master uses a slave's NAD and its `message_ID` to assign the unique protected identifier to each slave. Each `published` frame and all those `subscribed` frames (with matching data) will receive the same protected identifier. A slave will only respond (with data) to those frames sent by the master that include a protected identifier associated with one of its `published` frames. Even though a slave node will listen to all frames, it knows that only those frames with protected identifier's associated with one of its `subscribed` frames contains data that it can use.

The LIN bus transfers data using frames. The diagnostic frame was discussed as a way for the master node of a LIN cluster to have a dialog with individual slave nodes. Once all nodes have been initialized (received their individual assignments), the master can move into an Operational mode, where required communication is scheduled based on the information requested by the individual NCFs. Using the `max_period` and `min_period` declared by each frame and a `time_base` declared by the master, a schedule of frames is created (in the LDF) that satisfies the requirements of all the declared frames. The schedule is a list of frames (and delay times) that will be repeated. The delay times are based on the maximum time necessary to transmit the frame and receive the frame's response. Although this is fixed, it can be different for each protected identifier, based on the number of data bytes in the response. The delay time for that frame is the maximum frame time rounded up to the next even increment of the

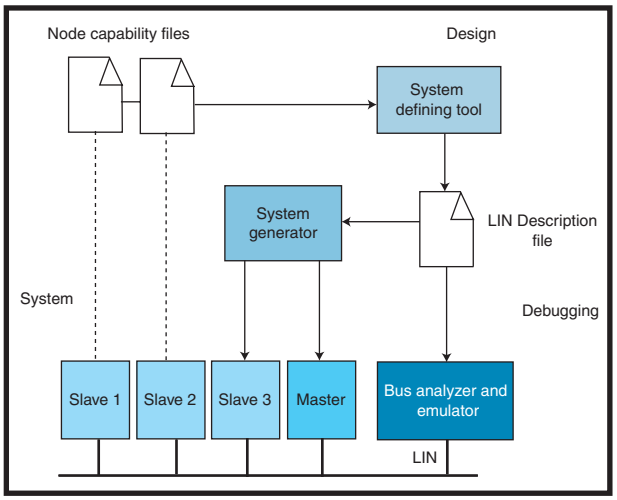


Figure 2—*This block diagram shows the LIN cluster split into three areas of design. The design phase begins with documentation. The system phase includes the actual cluster hardware. The debugging phase uses the documentation to test the actual hardware.*

time_base. In many configurations, this will simply be a list of the protected identifiers of all published frames. It may be necessary in larger clusters for some frames to be included more than once in a schedule list to assure that they meet their maximum and minimum requirements. To make scheduling a bit more flexible, multiple schedules can coexist, but only one can be active at any particular time.

I want to mention one of the zingers introduced into this otherwise straightforward protocol. It seems that if your cluster has data that is not expected to change very often, the frames will just continue exchanging this unchanging data. This is a waste, so let's complicate the protocol by allowing responses from multiple slaves at the same time, thoroughly destroying the original premise that only one slave can respond. The idea here is that a master can send a special frame that allows select slaves to respond and if no one responds, it can assume nothing has changed (and maybe go to sleep for a while). The problem now is that if multiple slaves respond, a bus conflict arises, and the response frame is useless. The master can then switch schedules using frames that guarantee a single response from each frame.

These special frames are called event_triggered_frames. Assigning the same protected identifier to multiple slaves creates them. Look back at the NCF in Listing 1. In the frames section, you will see an optional entry for event_triggered_message_ID. When a slave declares one of these, it allows the LDF to assign it a protected identifier that may be assigned to multiple slaves. (Only slaves with identical response lengths will be assigned the same protected identifier.)

Listing 1—*The NCF consists of the basic information necessary to describe a slave node. Its ASCII-based format allows it to be used to input to either a manual or an automatic LDF design process. Note that blue = choice and bold = optional.*

```
node_capability_file;
LIN_language_version = (ASCII string);       (0.01-99.99)

node (ASCII string) {                        (node name)
   general {
   LIN_protocol_version = (ASCII string);    (0.01-99.99)
   supplier = (integer);                     (supplier id)
   function = (integer);                     (function id)
   variant = (byte);                         (variant id)
   bitrate = (integer);                      (1000-20000 bps)
     or select (integer), (integer);         (acceptable rates)
     or automatic min (integer) max (integer);(acceptable rate)
   volt_range = (real), (real);              (integer or decimal)
   temp_range = (real), (real);              (integer or decimal)
   conformance = (ASCII string);             (LIN2.0 or none)
}
diagnostic {
   NAD = (integer), (integer);               (1-127)
   P2_min = (integer) ms;
   ST_min = (integer) ms;
   support_sid  { (byte), (byte)};           (supported sids)
   max_message_length = (integer);           (buffer size)
}
frames {
   publish (ASCII string) {                  (frame name)
   or subscribe (ASCII string) {             (frame name)
     message_id = (integer);                 (0-0xFFFF)
     length = (byte);                        (1-8)
        min_period = (integer) ms;
        max_period = (integer) ms;
        event_triggered_message_id = (integer);
        signals {
           (ASCII string) {                  (signal name)
              init_value = (interger);
              size = (interger);
              offset = (interger);
              encode (ASCII string) {        (encoding name)
                 logical_value, (integer),   (signal value)
                 (ASCII string);             (label)
                 physical_value (integer),   (min value)
                           (integer),        (max value)
                           (real),           (scale)
                           (real),           (offset)
                           (ASCII string);   (label)
                           bcd_value, (bcd); (signal value)
                      ascii_value, (byte);   (signal value)
                   }
                 }
              }
           }
        }
     status_management {
     error_response = (ASC string);
                                             (signal name)
   }
   free_text {
     (ASCII string);                         (description)
   }
}
```

API

Specifications for the LIN 2.0 are based on tools presently available to help the LIN bus designer achieve a working system with minimal effort. Figure 3 shows how these high-end tools manage the development process. The application program interface (API) is a collection of routines written in C to handle the master and slave processes necessary for LIN 2.0 conformance. Depending on implementation, a node may not require a LIN software device driver. The LIN 2.0 API provides a driver that is implemented by the LIN tools to maintain standardization. API functions can be used as guidelines if you will be programming in a language

PROTEUS

ELECTRONIC DESIGN

Version 7 Available NOW!

* New User Interface.
* New Design Explorer.
* New 3D Visualization.
* New Simulation Advisor.
* New Diagnostic Tools.
* PIC24 Simulation Available.

FROM CONCEPT



TO COMPLETION

SCHEMATIC CAPTURE | PROSPICE | EMBEDDED SIMULATION | PCB DESIGN

ISIS SCHEMATIC CAPTURE

A powerful capture package tailored for todays engineer and designed to allow rapid entry of complex schematics for simulation and PCB Layout.

PROSPICE MIXED MODE SIMULATOR

A customised implementation of the industry standard Berkeley SPICE 3F5 engine with extensive optimisations and enhancements for true mixed mode simulation and circuit animation.

VSM VIRTUAL SYSTEM MODELLING

The worlds first and best schematic based microcontroller co-simulation software. Proteus VSM allows you to simulate the interaction between software running on a microcontroller and any analog or digital electronics connected to it. This streamlines the project lifecycle and obviates the need for expensive hardware analysis tools.

ARES PCB DESIGN

A modern and professional layout package which seamlessly integrates with the ISIS capture software. Features such as autoplacement and autorouting, interactive DRC and an intuitive interface all serve to maximise productivity and reduce time to market.

LABCENTER ELECTRONICS LTD.
A technology pioneer in the EDA industry since 1988.
Technical support direct from the program authors.
Flexible packages and pricing tailored to customer requirements.

CONTACT US NOW
to discuss requirements or request a **FREE** evaluation copy.

R4 Systems Inc.
1100 Gorham St
Suite 11B-332.
Newmarket Ont.
Canada L3Y 8Y8

Authorized Distributor of Proteus
Other products include:
Proton + Development Suite
Discovering PIC Books and Kits
PIC Educational Courses

Toll Free 866.499.8184
Tel: 905.898.0665
Fax: 905.898.0683
email: info@r4systems.com
www.r4systems.com

KEIL™
An ARM® Company

Microcontroller Development Tools

Only 4 Steps...

...are required to generate efficient, reliable applications with the µVision IDE and development tools from Keil.

Step 1. *Select Microcontroller and Specify Target Hardware*

Use the Keil Device Database (**www.keil.com/dd**) to find the optimum microcontroller for your application.

In µVision, select the microcontroller to pre-configure tools and obtain CPU startup code.

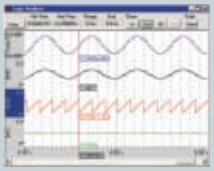
Step 2. *Configure the Device and Create Application Code*

The µVision Configuration Wizard helps you tailor startup code to match your target hardware and application requirements.

Extensive program examples and project templates help you jump-start your designs.

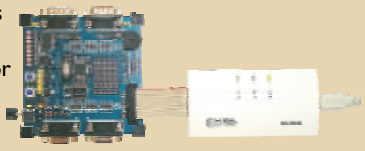
Step 3. *Verify Program Execution with Device Simulation*

High-speed simulation enables testing before hardware is available and helps you with features like instruction trace, code coverage, and logic analysis.

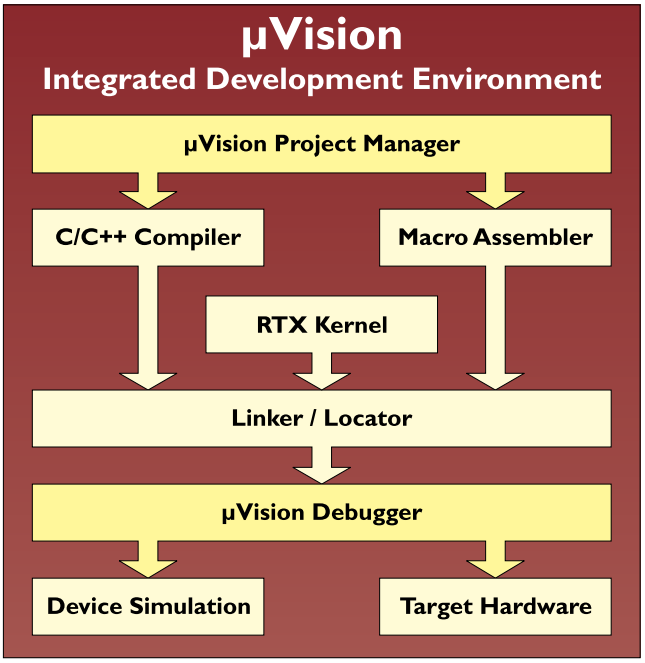


Step 4. *Download to Flash and Test Application*

Once your application is runs in simulation, use the Keil ULINK USB-JTAG Adapter for Flash programming and final application testing.



Keil Microcontroller Development Tools help you create embedded applications quickly and accurately. Keil tools are easy to learn and use, yet powerful enough for the most demanding microcontroller projects.



Components of Keil Microcontroller Development Kits

Keil makes C compilers, macro assemblers, real-time kernels, debuggers, simulators, evaluation boards, and emulators.

Over 1,200 MCU devices are supported for:

- **8-bit** - 8051 and extended 8051 variants
- **16-bit** - C166, XC166, and ST10
- **32-bit** - ARM7, ARM9, and Cortex-M3

Download an evaluation version from
www.keil.com/demo

800-348-8051 **www.keil.com**

other than C.

A separate diagnostic API is included (not to be confused with the diagnostic frames mentioned above). These functions make the LIN bus accessible via a CAN-to-LIN interface running on a CAN bus.

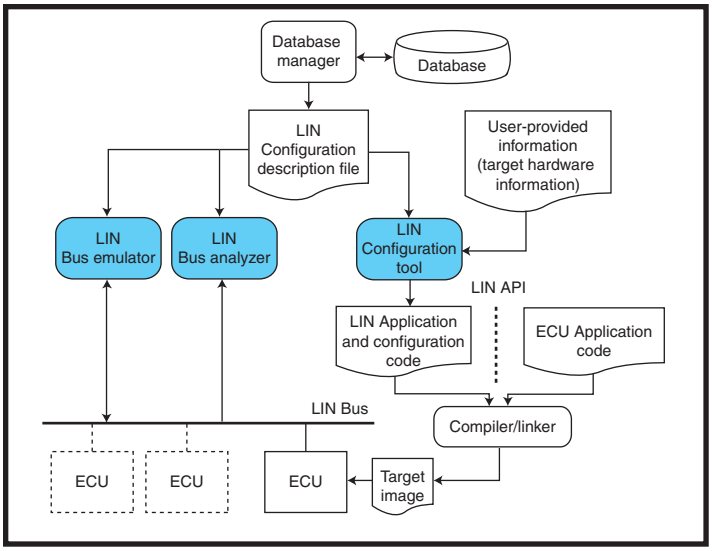


Figure 3—*The tool's environment (developed to support the LIN standard) can automatically produce code based on your design documentation to help automate the entire design process.*

DIAGNOSTICS

The diagnostic frames described earlier, which are used to configure NAD and assign protected identifiers, are covered under node configuration in this section. The diagnostics having to do with the CAN-to-LIN bus are not thoroughly discussed here and do not directly pertain to nodes (or clusters for that mater) that are by themselves separate entities. So, I leave this up to you to investigate if you are interested.

You're down to the frame structures themselves now. The master uses a schedule list to determine which frame will be transmitted and delayed until the next frame (frame slot). Thus, the frame being sent is shorter than the actual frame slot. The frame contains a header, a response space, and a response (see Figure 4). The master provides the header consisting of a break, a synch byte, and the protected identifier. The response space is dead time, where the master has finished sending the header and the slave has not yet begun its response. The slave's response is fixed (declared by the NCF) at 1 to 8 data bytes. The slave also includes a checksum byte. LIN 1.3 specifies that the checksum include only the data bytes. LIN 2.0 specifies that the protected identifier be included along with the data bytes in calculating the checksum. (That's one reason the NDF includes the protocol version number.) The checksum is calculated by adding individual bytes. If any sum produces a carry, the carry is added to the sum. The final byte is complimented.

All bytes are transmitted as a serial 10-bit word with a start bit, 8 data bits, and a stop bit except for break. The break has at least 13 dominant bits (including the start bit) followed by the break delimiter (stop bit). The protected identifier is actually a 6-LSBit identifier with a 2-MSBit-parity protection. Exclusive ORing bits 0, 1, 2, and 4 form bit 6. Taking the complement of the exclusive ORing of bits 1, 3, 4, and 5 forms bit 7. A protected identifier of 0x01 translates to a byte field of 0xC1 and a protected identifier of 0x03 translates to a byte field of 0x03. So, the byte value used to transmit a protected identifier is actually where the protection comes in.

Although a master has network configuration tasks, its main task is to manage network communication. The behavior for this task is to wait for the next time slot and then issue a break, synch, and protected identifier of the next frame on the schedule. If by chance the master also has an internal slave node (as many do), it must also behave like a slave node; otherwise, it merely waits.

A slave's behavior is much more complicated. It may be dormant (asleep) or active (awake). Even in a dormant state, a node's receiver is active and will wakeup the node if a break and synch is detected. Now the node is looking for a protected identifier that it recognizes. If none is found or the parity is incorrect, the slave will go back to a dormant state. If the protected identifier matches a `published` frame, the slave will transmit its data and checksum. If the protected identifier matches a `subscribed` frame, the slave will receive the data and checksum, using the data if the checksum is correct. There has been only one mandatory error flag (bit) that must be made available to the master. The `response_error` status bit indicates that an error in the response field is detected by the slave (read back).

PHYSICAL LAYER

A master's clock must be accurate to 0.5% of the nominal data rate being used. Slave nodes can use less expensive (accurate) system clocks (up to 14% error) as long as they can self-adjust based on the synch byte transmitted following a break. Since the synch byte must be a 0x55, the resulting pattern is alternating dominant (beginning with the start bit) and recessive states can be timed by a slave node. By measuring the time between the first falling edge and the fifth falling edge and then dividing by eight to get bit time, a slave can adjust its data-rate generator to

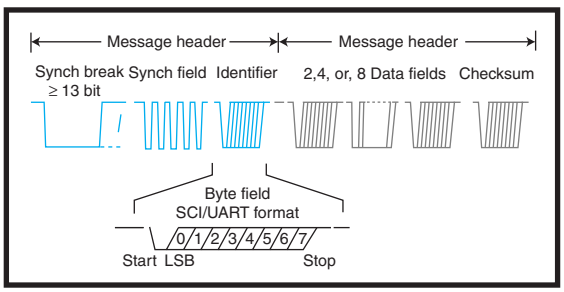


Figure 4—*The LIN frame structure contains a header provided by the master node and a response provided by a single slave node. All of the frame data (except the break) takes on the familiar 10-bit serial format of 8 data bits, surrounded by a start and stop bit. All of the bits are sent least significant bit first. All data is sent least significant byte first.*

improve accuracy. The EUSART peripheral in some Microchip Technology PIC microcontrollers has auto baud and send/recognize break functions.

The LIN bus consists of a +V (battery), ground, and a single bidirectional communication path. A LIN transmitter/receiver isolates itself using protection diodes, so the bus voltage or node voltage cannot feed back onto the LIN bus's power if the battery is lost (see Figure 5). Therefore, the internal supply voltage (V_{SUP}) and communication voltage swing will be somewhat less than the LIN bus battery voltage. The dominant (falling to 40% of V_{SUP}) and recessive (rising to 60% of V_{SUP}) thresholds are based on the internal supply (which is based on the LIN bus's battery voltage). The LIN 2.0 line characteristics limit the LIN bus length to 40 m and 16 nodes.

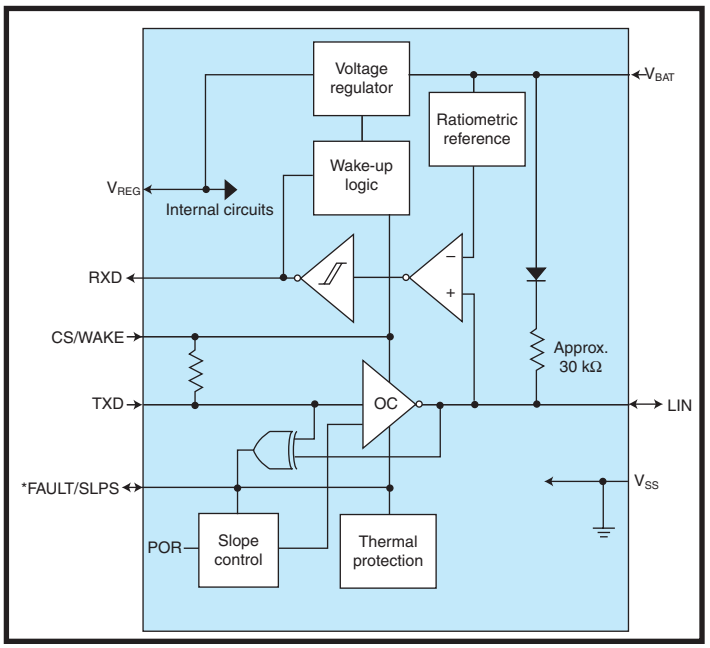


Figure 5—*The Microchip Technology MCP201 LIN transmitter/receiver protects the LIN bus by disconnecting the transmitter if a problem exists. In Low-Power mode, LIN transceivers continue to monitor the LIN bus for minimum-power designs.*

IMPLEMENTATION

I designed a small circuit to allow data to be passed by using the LIN bus protocol (see Figure 6). The circuit can be used as either a master or a slave node. I began by creating an NCF for a slave node and an NCF for the slave portion of a master (that's right, the master can also serve as a slave node). This allowed me to put together an LDF for the project. Since I was not using off-the-shelf products, I was able to write the application code in such a way that no nodes needed reassignment. In other words, I didn't need to include those functions to reassign NADs, assign `frame_IDs`, or any other configuration overhead. The code is therefore straightforward and just has to deal with sending and receiving actual data. The schedule has two unconditional `frame_IDs`, 0x00 and 0x01. `Frame_ID` 0x00 requests 3 bytes of

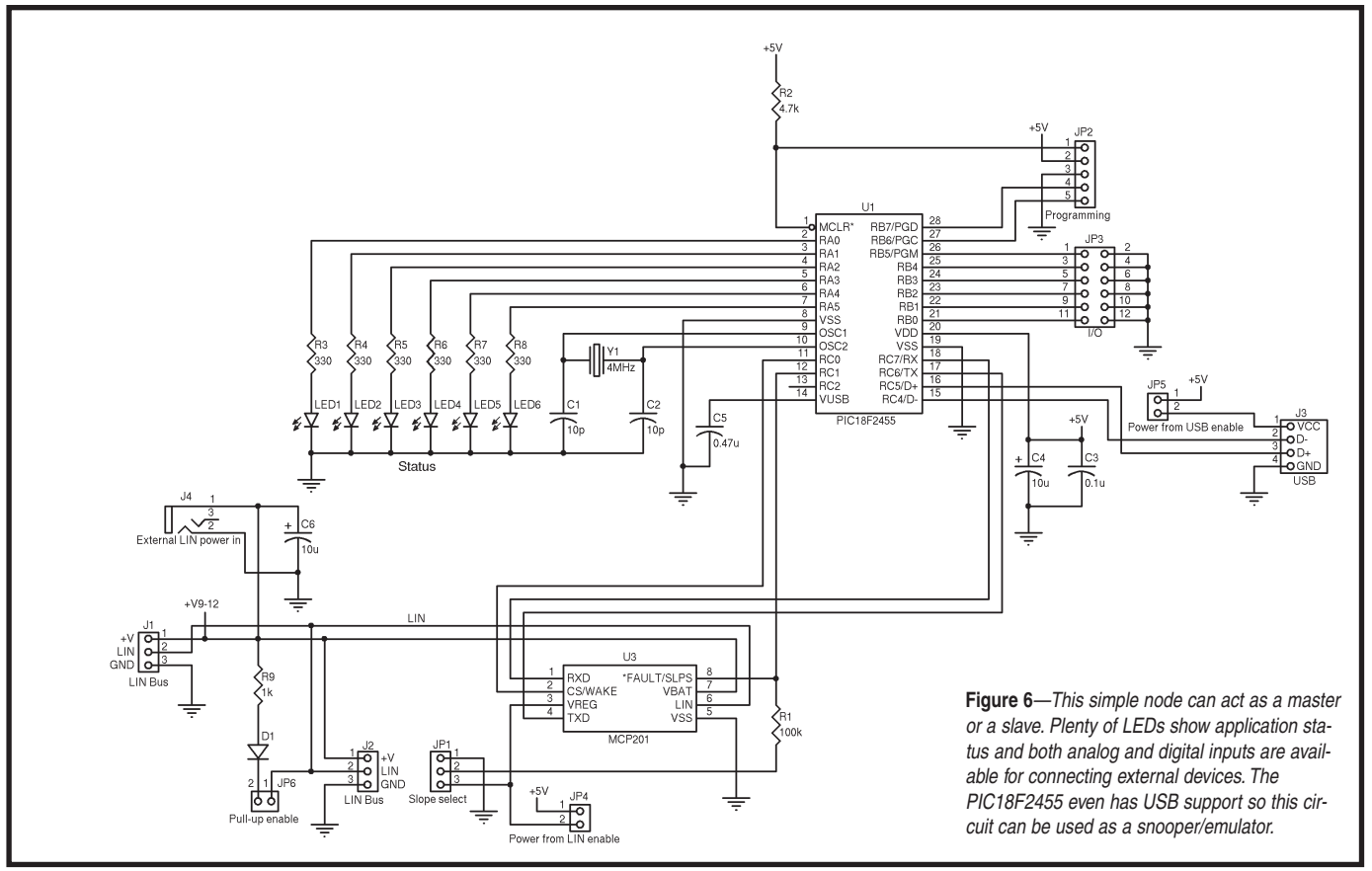


Figure 6—*This simple node can act as a master or a slave. Plenty of LEDs show application status and both analog and digital inputs are available for connecting external devices. The PIC18F2455 even has USB support so this circuit can be used as a snooper/emulator.*

data from the master's slave portion, while `frame_ID` 0x01 requests 4 bytes of data from the slave node.

The long-range project will include at least one more slave node. This node will be some kind of display that will make use of the data transmitted by the first two nodes used in this project. Presently, there is no reason why adding this third node would necessarily change the programming of the first two nodes, because it will only be listening to the network (much like a LIN snooper node might). However, under LIN specifications, its NCF would be included in the creation of the cluster's LDF because the master would need to assign it `frame_IDs` to watch for. In reality, since I am in control, I can write the application for this node using the `frame_IDs` 0x00 and 0x01, and everything will work well. Although communication between nodes follows the LIN 2.0 specifications, this project's application does not need or support reconfiguration and diagnostic functions.

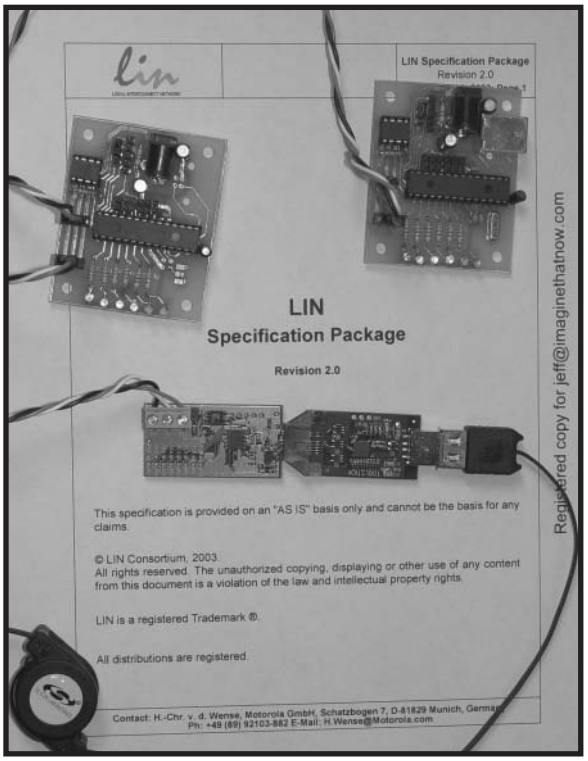


Photo 1—*I configured identical PCBs as master and slave nodes. I then daisy-chained a ToolStick to the cluster to monitor LIN traffic.*

Photo 1 shows the prototype PCBs (a master node and a slave) daisy-chained to a third node. This third node is a Silicon Laboratories ToolStick with a LIN DC daughter board. Available demo software (a LIN snooper application) can be downloaded to the Stick. The LIN snooper application uses a PC to display information about LIN bus traffic (see Photo 2).

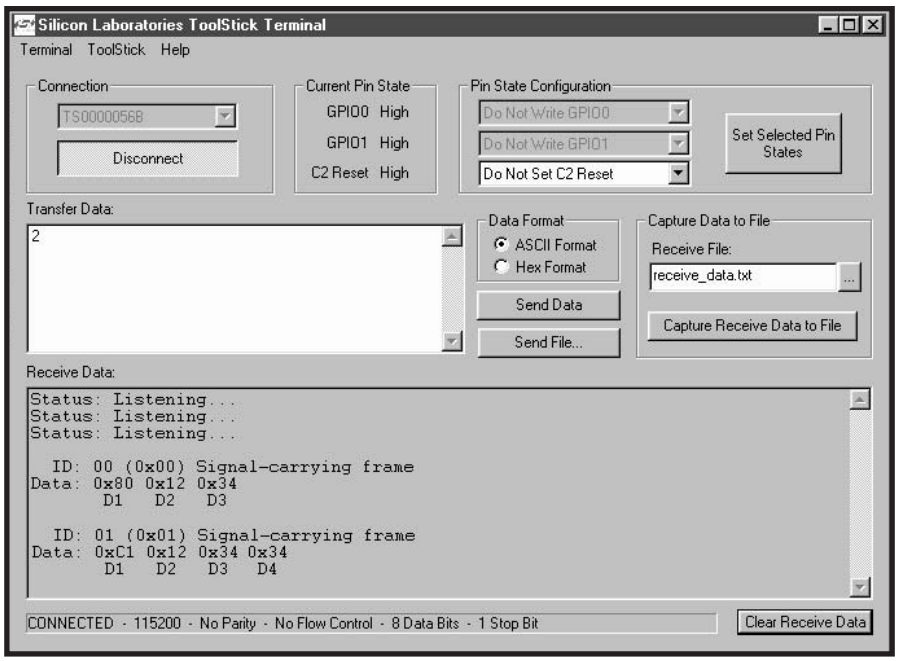


Photo 2—*The Silicon Laboratories ToolStick has a demo application for monitoring the LIN bus. The ToolStick Terminal will display what it sees on the LIN bus. In this case, the master asks for `frame_ID` 0x00 and it responds with 3 bytes of data (actually 4 bytes are transmitted: the 3 data bytes and a checksum). The master then asks for the `frame_ID` 0x01 and the slave node responds with 4 bytes of data.*

REALITY

You have to search hard for manufacturers that have adopted the LIN bus in their automobiles. Certainly the Europeans are well ahead of U.S. automakers embracing the whole LIN-bus idea. You won't find the LIN bus listed in the specifications of vehicles. What consumer cares about the skeletal design of their car? Maintenance-free is what we desire and we don't care how it's done. Gone are the days of tinkering with an engine on weekends. Today, if you don't have the diagnostic tools, forget it. Automotive problems have become just a number dumped by some computer. This means that whatever bus is being used is useless without the documentation that defines the data. How nice is it for that documentation to begin with the original design and to be an integral part of it right through production?

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

RESOURCE

LIN Consortium, www.lin-subbus.org/frontend/stylesheets/request_doc.htm.

SOURCES

MCP201 LIN transceiver and PIC18F2xxx Microcontrollers
Microchip Technology, Inc.
www.microchip.com

ToolStick
Silicon Laboratories, Inc.
www.silabs.com



USB: (Wire)Less is More

Tom covers new developments in wireless power technology, tells the story of ultra-wideband technology, and cuts through all of the "wireless" USB hoopla. Will you be the next designer to bring wireless power to the masses?

In last month's column about the Cypress Semiconductor Express visual programming language for PSoCs, I mentioned their "Wireless USB" offering. Subsequently, I received some wireless USB hardware courtesy of a third-party supplier, Artaflex. Their Falcon module is based on the Cypress CYRF6936, a complete 2.4-GHz radio on a chip. Though it didn't make the deadline for last month's story, I do want to report that I checked out the gear with a simple, but personally useful, application.

The way my house is laid out, you have to either go up and down a flight of stairs or step outside to see if the garage door is open. Kind of a pain in the keister, right? Using Express and the Artaflex board, it is easy to whip together a wireless door position sensor (see Photo 1). One unit mounted on the garage door uses a low-g accelerometer as a tilt indicator to radio that information to another unit in the living quarters that lights a red LED for "open" and a green LED for "closed."

The follow-up serves as a segue to this month's column which isn't about "wireless USB." It's about "wireless USB." Needless to say, a little explanation is in order.

USB UNPLUGGED?

Cypress may call their proprietary radio scheme "wireless USB," but I'd say it's more like "lesswire" rather than "wireless." Consider the example of the wireless keyboard or mouse that many of us use with our PCs. A typical setup puts a radio transmitter in the I/O device (i.e., keyboard and/or mouse), which communicates wirelessly to a receiver that's a standard USB device connected by cable to the PC's USB port.

The beauty of this approach is that the PC doesn't know or care that the keyboard or mouse is wireless, since at the end of the line there is indeed a wire (or dongle) plugged into the PC's USB port. Thus, it's all blissfully transparent to the OS and application software. Another thing to note is that the particular details of the radio used don't matter at all. Since the transmitter (in the I/O device) and receiver (plugged into the PC) are sold as a matched pair, any radio scheme that gets the bits from here to there works fine.

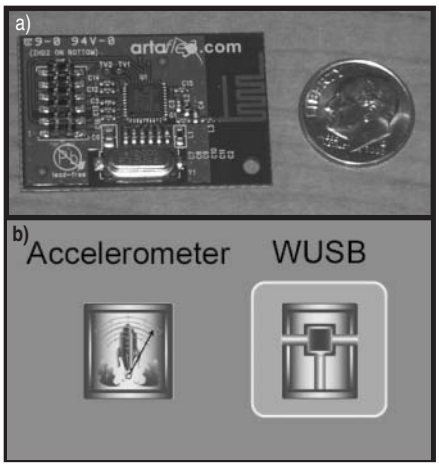


Photo 1—*A two-line program usually can't do much. But using a low-g accelerometer and a radio module from Artaflex* **(a)**, *all it takes is a "two-object" Express program* **(b)** *to implement a wireless tilt sensor.*

This type of "Lesswire USB" lashup is pretty much limited to simple, low-speed devices. Walk the aisles at your local PC shop and you'll find plenty of wireless keyboards and mice, but you won't find wireless disks, printers, cameras, MP3 players, and so on. That's because upping the ante in terms of performance and device complexity stresses the differences between a radio and a wire to the point that they become obvious and problematic.

There are a lot of wired versus wireless issues, such as ease-of-use, security, and reliability, that we'll get into a little later. In terms of the "Lesswire USB" approach typified by today's wireless mouse or keyboard, there's only one issue that matters—speed, or rather a lack of it—and it's a showstopper.

For instance, the Cypress solution is only good for 1 Mbps with a tailwind and lowlier keyless remote-class (e.g., 27 MHz) radios deliver even less. That works for a mouse or keyboard (or a simple application like my garage door position sensor), but not much else.

GO WIDE YOUNG MAN

Ultra-wideband (UWB) is a topic that's been around for some time, but frankly until now it has been more sizzle than steak. That's all about to change, with real UWB chips and boxes hitting the shelves even as we speak (see Photo 2).

From 50,000', the UWB concept is pretty simple. Instead of hitching a

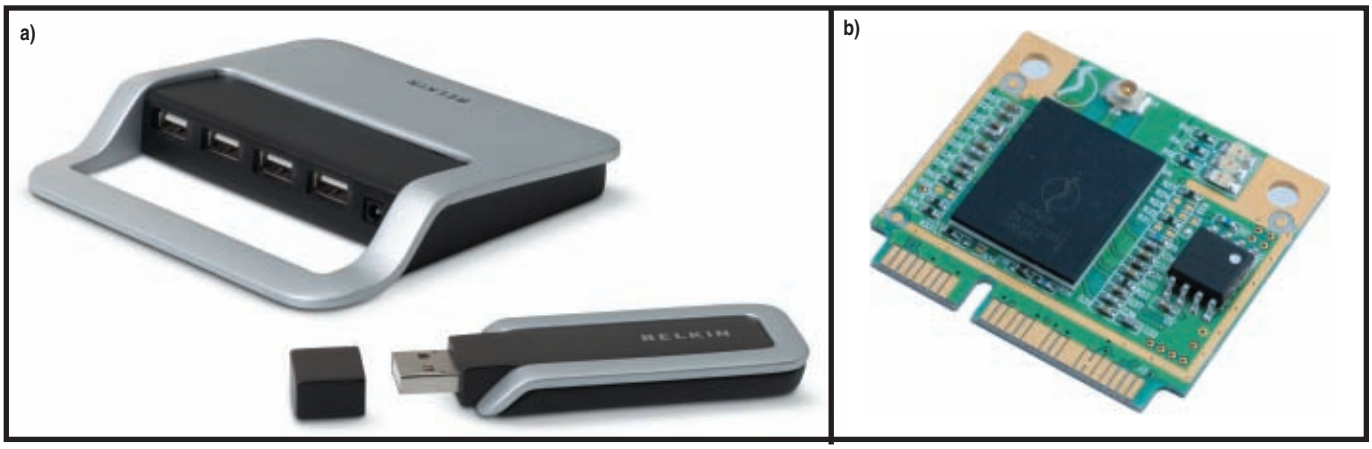


Photo 2—*The market for wireless USB gadgets is poised for takeoff in 2007. Belkin false-started with a "Cablefree" USB hub* **(a)**, *but they will be switching to Certified Wireless USB using Intel silicon shortly. Booming business prospects are also enticing new players into the fray, such as Staccato Communications, who offers Certified Wireless USB reference designs based on their "Ripcord" UWB chipset* **(b)**.

ride on sine waves, like a typical "carrier-based" radio, UWB uses discrete pulses to pass information. There are a lot of ways to do that, for instance by varying the pulse timing, amplitude, polarity, and so on. But the UWB bottom line is pulses instead of waves.

The good news is that UWB can deliver a lot of throughput, even using conventional (i.e., complementary metal-oxide-semiconductor (CMOS)) technology. The bad news is the short and frequent pulses generate harmonics that almost literally cover the spectrum, raising the prospect of drowning out every other wireless gadget out there. What to do?

The answer, as blessed by the FCC, is pretty simple: limit the allowed power such that a UWB radio is no more obnoxious than the typical unintentional emitter (i.e., the familiar FCC Part 15 limit). The combination of low power spread across a huge spectrum means that within the relatively narrow bands used by existing (i.e., carrier-based) radios, a UWB signal appears as little more than a slight increase in background noise (see Figure 1).

While UWB, like all communication schemes, can make trade-offs between speed (i.e., bandwidth) and distance (i.e., SNR), the low-power limitation makes UWB a good fit for so-called personal area networks (PANs) with range measured in ones to tens of meters. For PAN applications, limited range is, in fact, a virtue that emphasizes the "personal" part of the equation by minimizing interference between your PAN and mine. Not to mention all the other LANs, WANs, etc. crowding the spectrum.

PHY VS. PHY

FOR IMMEDIATE RELEASE

Belkin CableFree USB Hub Enables Instant Wireless Connectivity of USB Devices

Powered by Freescale's Ultra-Wideband technology, Belkin brings the world's first wireless USB Product to the market

(Los Angeles, CA)—Jan. 3, 2006—Belkin announces its new CableFree USB Hub, the industry's first USB Hub that does not require a cable to connect to the computer.... Beginning in early spring, (Updated 10/06: Ship date change to mid-December 2006), Belkin will offer...*

*...*Because the original chipset manufacturer for our CableFree USB Hub has stepped away from the CableFree USB business, we've had to change to a different chipset. This has caused us to push back the availability of the Hub to September. Please look out for more details in a coming press release.*

—Belkin Press Release, January 3, 2006

The UWB story gets weird at this point. About five years ago (December 2002), the IEEE decided the time was right for UWB and authorized a standard effort (IEEE 802.15.3a), fulfilling their traditional role as mediator of all things networking. Err, "mediator" may be optimistic. I'd say "marriage counselor" is more like it.

Sure enough, the 802.15.3a effort quickly devolved into a "War of the Roses" between two camps, respectively led by historic archrivals Intel and Freescale Semiconductor (then Motorola).

On the surface, the argument was ostensibly about arcane aspects of the modulation scheme, namely Direct-Sequence (DS) championed by Freescale and Orthogonal Frequency Division Multiplexing (OFDM) by Intel. Technical issues provided the excuse for arguing, but it was

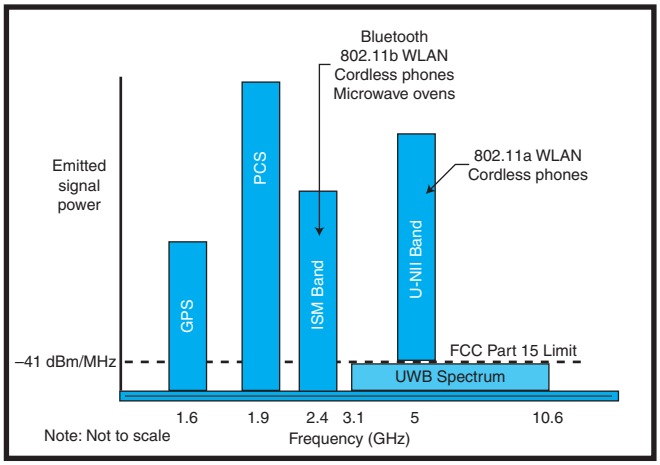


Figure 1—*According to the FCC, anything that hogs more than 500 MHz of spectrum qualifies as ultra-wideband (UWB). To prevent interference, the rules limit UWB transmit power to a level so low (the familiar FCC Part 15) as to go unnoticed by existing spectrum users.*[1]

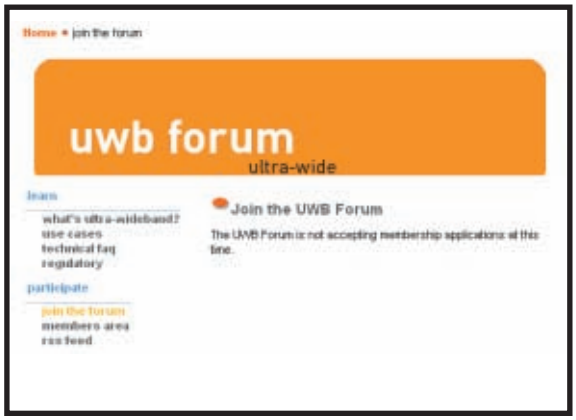


Photo 3—*The great UWB modulation war between direct-sequence (DS, backed by the UWB Forum) and orthogonal frequency division multiplexing (OFDM, backed by the WiMedia Alliance and the USB Implementers Forum) is finally over, and DS lost.*

Fighting against your PCB-Design Software?

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

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

EAGLE 4.1

Schematic Capture • Board Layout
Autorouter

for Windows® Linux® Mac®

Now available for Mac!

Version 4.1 Highlights

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

FREE

EAGLE 4.1 Light is Freeware!

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

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

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

Pay the difference for Upgrades

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

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

always really about NIH and trying to preemptively seize high ground in the marketplace.

The jockeying for position and bad-mouthing went on and on for years, finally reaching a bitter deadlock. Ultimately, in January 2006 the two camps finally found something to agree on, namely that they would never agree, and voted to dissolve the UWB standard. At this point, I think another thing everyone can probably agree on is that the IEEE standards process, held hostage to ballot stuffing and paralysis-by-analysis, probably needs some work.

Anyway, the saga continues. The corpse of 802.15.3a wasn't even cold before the Freescale and Intel camps split into their own consortia under the banners of the "UWB Forum" (DS) and "WiMedia Alliance" (OFDM), respectively. With no more fussy IEEE details (like "rules" and "voting") in the way, the battle became more tactical. Instead of speechifying about technical trinkets and backroom arm-twisting, each side quickly mobilized its forces for a double-time march to market.

Here's where USB comes back into the picture. Keep in mind that UWB is about a future that goes beyond USB and PCs, including everything from entertainment systems in the living room (i.e., wireless A/V) to vehicular radar on the road ("Look Ma, no hands—on the wheel"). But these bluer-sky apps will take time to emerge and those markets have a lot of inertia to overcome.

By contrast, USB probably represents the first, and fastest, way for UWB to break out of the lab and onto shelves. Compared to the living room or road, the PC market evolves and adopts new technology at a decently fast pace. Everyone has seen how quickly the wired-USB market was able to migrate from USB 1.0 to 2.0. Wireless USB was understood to be the first "killer" UWB application, and thus became the proxy UWB battleground.

As a founder of the original USB consortium, Intel naturally aligned with the "certified" wireless USB initiative. That left the Freescale camp to

We search the world for time-saving, easy-to-use state-of-the-art products!

Testing

LCD Scope

NEW Low-Cost!



PDS5022S - 25 or 60MHz 2-ch + trigger standalone bench scope with USB connection and 7.8" color LCD.

PDS5022S	**Sale! $329**
PDS6062T	**Sale! $579**

Handheld Scope

NEW



HDS1022M - 20MHz 2-ch standalone scope and DMM with USB connection and 3.8" color LCD.

HDS1022M	**Sale! $499**

Mixed-Signal PC Oscilloscope



100 MHz Scope and Logic Analyzer lets you do complex triggering to find hard-to-get-at glitches, spikes, etc. Huge 4 or 8 MS buffer for deep data drilling and zooming. Optional built-in swept signal generator. 2 Analog channels x 10, 12, 14 bit with more than 60 dB dynamic range. 8 digital inputs for mixed signal display/trigger.

CS328A	**$1149**
CS700 (signal generator)	**$249**

PenScope



PS2105 - 25MHz USB powered scope-in-a-probe! Up to 100MS/s, 24kS buffer C/VB/Delphi/LabView/VEE drivers.

PP315	**$349**
PP317 (10MHz, 50MS/s)	**$199**

Logic Analyzer

NEW



ZeroPlus - Intuitive full-featured 16-channel with 4mb memory. Sophistcated triggering. I2C, SPI, and UART monitoring. Includes clipleads, USB cable and software.

LAP-16128U	**$399**

I2C

USB to I2C



USB I2C/IO - Provides a simple "drop-in" solution for connecting your PC to 90kHz I2C + 20 I/O lines. Free software. Use multiple boards for more I2C/IO.

USB I2C/IO	**$79**

STOP THE PRESS!

- NEW PRODUCTS ARE ARRIVING EVERY DAY!

Please go to **www.saelig.com** to see late-breaking announcements.

Ethernet to I2C



FMod-TCP - TCP/IP server with I2C, 19 x digital I/O, 5 x 10-bit A/D inputs, and UART serial interface. Connect I2C devices/sensors to an Ethernet network!

FMod-TCP	**from $232**

I2C for PCs



PCI93LV: industry-standard I2C card for PCs. **WINI2C/PCI** software gives windows-interface to develop and debug I2C bus systems. **UCA93LV** is new USB version. ***NOW*** - transparently monitor at 400kHz!

	$499

FREE COFFEE



Call 1-888-7SAELIG

Mention Offer# SBW

to get a free Starbucks Card with your order!

While supplies last - not available with any other offers

Ethernet to I2C

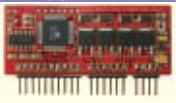


NM7010A-LF REV 2 - Complete, hardwired TCP/IP-I2C module. Offloads stack for hi-speed Internet-enabling microprocessor systems. Drop-in W3100A-based board for instant network speeds to 8Mbps.

	$22

I/O

Stepper Motor



Easy-Step 3000 - Advanced stepper motor drive and control system for unipolar stepper motors up to 35V/3A - ideal for robots/industrial control - turns you into an instant motor control expert!

	$119

Signal Wizard



Signal Wizard - easy-use real-time DSP-based filter board for audio bandwidth signals. Design filters in *seconds* without any DSP knowledge!

Signal Wizard v2.5	**$699**

USB to 24 x I/O



USBI/O24R - 24 latched I/O lines self-powered from USB port. Ubicom SX52 allows individual pin I/O programming. I/O control via PCs USB port.

USBI/O24R	**$69**

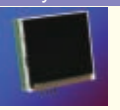
FATfile Storage



Micro ALFAT-SD - Ready-made way to easily add SD storage to your product. Serial data in - FATfile data out. **$39.95** Or design-in a micro ALFAT IC for MB storage.

	$14 ea.

Easy LCD



Micro LCD - Easy way to add 2" sq 128 x 128 color LCD to your project. Add text/graphics/animations for any micro with serial or USB.

Micro LCD MK II	**$81**
Micro LCD-OLED	**$99**

Touch-Sensing ICs



Quantum ICs - World's best capacitive sensor ICs for switching & control. Patented Adj. Key Suppression, Spread-spectrum, adj. sensitivity. In use worldwide by the world's largest consumer/appliance mfrs

	$1

Ethernet

Serial-Ethernet Cable



28" cable links serial devices to network without a PC, providing an instant IP address. RS-232 port compatible with most 10/100mpbs Ethernet hubs. Network your serial product easily!

eCOV-110-P	**$99**

Ethernet to Serial



EG-SR-7150MJ - Ready-to-go serial-to-Ethernet Gateway. Simple set-up for 10/100Mbps Ethernet - 230kbps serial. Ethernet/Internet-enable any serial part.

EG-SR-7150MJ	**$35**

Ether-IO



Ether-IO - UDP/IP-controlled 24 X digital I/O board. 3 x 8-bit TTL ports each independently programmable. Connects to any TCP/IP Ethernet network.

Ether-IO 24	**from $99**

Ethernet to I2C



FMod-TCP - TCP/IP server with I2C, 19 x digital I/O, 5 x 10-bit A/D inputs, and UART serial interface. Connect I2C devices/sensors to an Ethernet network!

FMod-TCP	**from $232**

TCP/IP IC



W3150A+ - Hardwired TCP/IP stack - cheap micros can now have internet capabilities at full speed! 10/100 Off-loads TCP/IP burden for fast time-to-market. Ethernet-enable almost ANYTHING!

	$7.20

Motion Control



PIC-Servo - Motion control boards with integrated power amplifiers for DC-servo/brushless-servo/stepper-motors. Easy connection to std serial ports for powerful multi-axis, mixed-motor control.

	$160

RF

Bluetooth



Emxys PearlBlue - 100m-Bluetooth module (UART master/slave bridge). Easy connection for micro's to PCs, PDAs, phones, etc. IceBlue - PIC16F877 based emb. dev. system for shortest Bluetooth development.

	$119

RFID Reader IC

NEW



RFID Reader Chip - Reads FIVE Int'l Stds ISO14443 A/B, Tag-IT, Felica and ISO15693. Single-chip solution.

TRH031M	**$6.95**
RSK-100 Eval Kit	**$350**

easyRADIO



easyRADIO™ Transceiver Modules make wireless data transmission simple for USA and Europe! Add wireless capability to your project today!

ER900TRS	**from $35 (100)**

Wireless Controllers



Owasys - GSM/GPRS/GPS/Bluetooth/Ethernet wireless controllers for M2M - for autos, machines, etc. RS232/485/CAN/A/D-I/O.

Owasys	**from $479**

TinyOne Zigbee

NEW



TinyOne Zigbee™ - Complete RF solution for fast and reliable data transmission. Miniaturized OEM RF module ensures fast and easy integration into wireless applications.

TinyOne Zigbee	**$55**

Data Modem



Data Modem - USB-powered radio transceiver boards add wireless RS232 to your device! Get remote control for your product from PC USB port!

RTM-USA	**$89**

USB

USB in one IC



FT232RL - FTDI USB1.1-serial IC with new "security-dongle" feature. Each IC has unique ID! Internal EEPROM, clockgen, pullups, etc.

FT232RL	**$4.05**

Check out the whole FTDI line!

FT232R Module



UM232R - FT232RL 24-pin DIP module - USB-TTL serial converter for your project. Reference circuit for FT232R. Plug-in module adds USB1.1 instantly!

UM232R	**$19.99**

USBwiz



USBwiz - Easily add USB peripherals to your system. Includes FATfile system for USB thumb & external USB hard drives. No USB knowledge needed. Plug-n-play!

IC $29.95	**Demoboard $59.95**

Easy USB



microUSB - Tiny ready-made USB Interface to 1 Mbps 0.7" x 0.5" based on SiLabs CP2101. Gnd/RX/TX Suspend, Reset, & 3.3V 100mA source.

microUSB	**$25**

GPS Logger



TrackStick - Pocket-sized logger tracks position data for vehicles, personnel, etc. Records date, time, location, route and speed. Shows data on Google maps!

PRO (vehicle powered)	**$239**
	$249

USB-Serial Adapter



FTDI - based serial port for laptops. Instantly adapts older RS232 products to USB! Included drivers provide virtual COM port via 45-inch USB cable.

USB-COM-S	**$24.95**

www.saelig.com

1-888-772-3544

1-888-7SAELIG

Saelig.com unique electronics

info@saelig.com

585-385-1768 fax

sensors expo & conference

REGISTER TODAY!

Conference: June 11-13, 2007 • Expo: June 12-13, 2007
Donald E. Stephens Convention Center • Rosemont, IL



Advances in Measurement, Detection, Monitoring & Control



The World of Sensing Technology

- **Measurement & Detection**
- **Wireless Sensing**
- **Systems & Embedded Intelligence**
- **Emerging Technologies & Applications**
- **Low-Power Sensing**





KEYNOTES



Steve Squyres, Ph.D

Roving Mars: *Spirit*, *Opportunity* and the Exploration of the Red Planet



Gregory W. Withee

Sensing the Health of the Planet Through the Global Earth Observation System of Systems (GEOSS)



Produced by:
QUESTEX MEDIA

Official Publication:
sensors

Gold Sponsors:
Bluetooth SPECIAL INTEREST GROUP
GE Sensing

For more information or to register, visit www.sensorsexpo.com or call 800-748-5045 and use source code 303M

come up with their "CableFree" USB flavor.

As with UWB itself, once again there were technical differences to bicker about. "Cablefree USB," courtesy of a little software sleight-of-hand, purported to enable wireless USB without requiring any software changes on the PC. Meanwhile, "Certified Wireless USB" called for more ambitious upgrades to the underlying protocols, changes that would require new OS drivers.

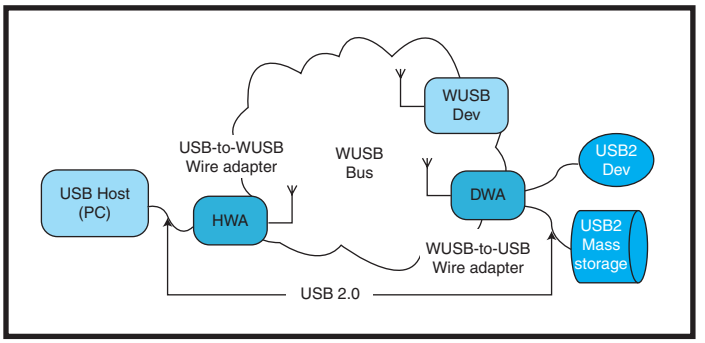


Figure 2—*The choice between wired and wireless USB isn't an either-or situation. Both will coexist using a combination of host (HWA) and device (DWA) adapters.*

And once again, the technical issues would be rendered semi-moot by a shakeup at the top. Preoccupied with their own corporate machinations (i.e., spinout from Motorola) and perhaps just conceding the inevitable, Freescale suddenly decided to hit the Reset switch on their UWB strategy. Adding insult to injury, they quit the UWB Forum, the very group they were instrumental in creating (see Photo 3).

Thus, '06 was a year filled with marketing fire drills and U-turns. There's Orwellian retro-editing of press releases (like the example from Belkin International at the start of this section) and "Product of the Year" awards for products that never were. Indeed, much of the history, admittedly not a pretty one, seems to have "disappeared" after the fact. Go to the IEEE standards web site and search for "802.15.3a" or the Freescale web site and search for "UWB." See what I mean? Maybe it was all just a bad dream and it never really happened.

DIAL "M" FOR MICROSOFT

As far as I'm concerned, when it comes to USB there is only one company that really matters, and that's Microsoft. It's not so much that Microsoft can single-handedly drive a particular solution. Surely they need aiding and abetting from the likes of Intel and PC suppliers. Rather, it's the recognition that nobody else can drive a solution without Microsoft's blessing. For example, consider a scenario that has Apple or the Linux crowd (or both together) trying to redefine and seize control of USB. It just won't happen. They wouldn't win, and they know they wouldn't win, so they won't even try. Microsoft may not be king when it comes to standards, but they are the kingmakers.

With Microsoft, Intel, and a 100-plus-member coalition of the willing behind it, I hereby declare "Certified Wireless USB" (I'll just use "WUSB" as shorthand) the winner. With that decided, let's see what the future holds.

First the basics. The WUSB specs require that all devices deliver at least 200 Mbps, with even higher speed (up to 480 Mbps) as an option. Note the implication of the high baseline speed; WUSB is overkill for low-end (i.e., mouse, keyboard) apps, which presumably continue to use the "Lesswire USB" approach.

Like wired USB, the protocol burden is asymmetric with the "host" doing the heavy lifting to make things easier (and thus lower cost and power) for "devices." However, it does incorporate dual-role capabilities to enable "no PC" apps, such as direct communication between a digital camera and a photo printer.

A single host in a WUSB network can support up to 127 devices. One big difference from wired USB is that WUSB uses a star point-to-point topology versus the "tree" of wired version. Thus, with one exception, there is no such thing as a WUSB "hub."

That exception involves easing the transition from here (i.e., wired USB) to there (i.e., WUSB). A new class of device is defined called a "Device Wire Adapter" (DWA). It can be considered kind of like a conventional USB hub with connectors and wires on the device side, but using WUSB for the upstream host connection (see Figure 2). A DWA allows folks to migrate their existing wired USB devices into the brave new world of WUSB.

The other half of the upgrade challenge is addressed with a "host wire adapter" (HWA) that has a wired connection on the host side and a WUSB on the device side. Just plug an HWA into an existing PC's USB port to enable the PC to function as a WUSB host.

Expect HWAs and DWAs to explode out of the blocks. It's going to be quite a business, since many or even most existing PCs will end up needing one

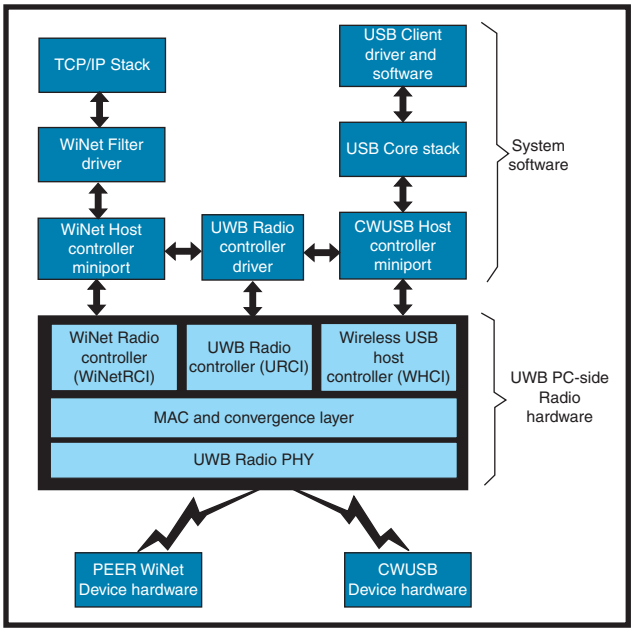


Figure 3—*Just as a single radio station may deliver, for example, both talk and rock, a single UWB radio in the PC of the future will support a variety of interfaces.*

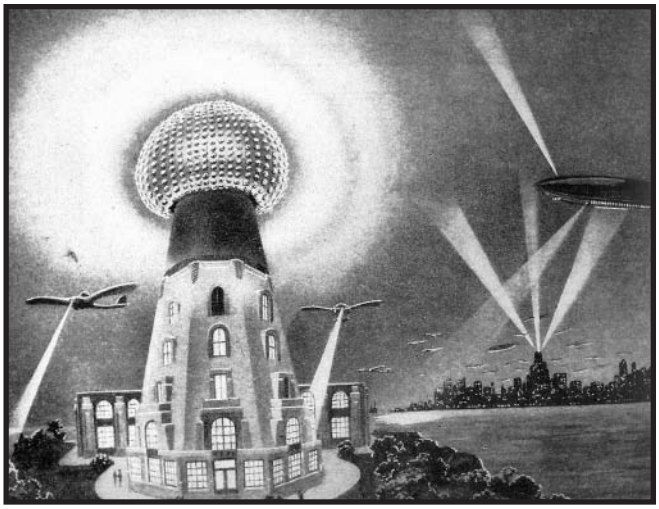


Figure 4—*The idea behind Nikola Tesla's turn-of-the-century Wardenclyffe Tower's centralized wireless power delivery was way ahead of its time—and still is. (Source: www.teslasociety.ch/info/galerie/bilder/G59.jpg)*

or both along the way.

One "ain't as simple as it sounds" aspect of wireless is the basic task of establishing a connection. With a wire, it's as fast and easy (and secure) as jacking in. The WUSB folks have put a lot of effort into delivering "wire-equivalent" out-of-the-box ease of use.

Ironically, one option is to rely on a wired USB port to establish an initial wireless connection. At first this sounds goofy, but maybe it's more practical than it seems. Even a WUSB-enabled PC is likely to come with wired USB ports for the foreseeable future. Similarly, I suspect many devices will come with both USB and WUSB connectivity.

The other approach, wireless-only, will have users push a button or some such to manually initiate a connection, much as with today's wireless keyboards and mice. However, the WUSB scheme is complicated by the need for bulletproof security. To prevent pirates from eavesdropping on, or even hijacking the connection, the user supervises and approves an initial key exchange. Subsequently, the keys are retained in nonvolatile memory so authentication and encryption of all WUSB traffic is automatic (and mandatory).

But above all, it's the reliability issue that really separates the wires from the radios. This is especially true for time-sensitive data (such as audio and video) and it is reflected in the significant changes WUSB makes to the isochronous device class.

Keep in mind the challenge is bigger than just WUSB. In a PC, a single UWB radio may carry a variety of "programming," including WUSB, Internet, and future versions of other popular interfaces (e.g., Bluetooth, 1394) as well (see Figure 3).

So, the host needs to be able to deal with uncertainty on both the supply (i.e., radio packet error rate) and demand (channel and load variation) side. That's not easy, and it requires a lot of new capabilities. For instance, the host needs the ability to dynamically change a device configuration (i.e., packet size vs. burst length vs. service interval) as overall circumstances dictate. Fortunately, most of the complexity resides in the host, so the folks designing PCs have to do the most head scratching. WUSB devices, notably isochronous ones, need to be smarter as well. But it's still generally the case that devices tell the host what they can do and ask for a certain level of service. It's up to the host to decide yea or nay and, if it is the former, deliver on the promise.

RETURN OF THE WALL WARTS

Of course, one irony of all the "wireless" USB hoopla is that it likely won't lead to less wires at all.

One of the best features of wired USB is the delivery of power. Take away the USB cable and you're left with either battery power or, more likely, a return of the wall warts (i.e., AC plug-in power supplies).

Fortunately, UWB is relatively power efficient (i.e., joules/bit) compared to other wireless schemes (e.g., 802.11), principally by virtue of the limited range. And the WUSB folks are well aware of the issue judging by the prominent role given power management in the spec.

On the other hand, the rosy scenarios call for UWB and WUSB to move lots of bits (MP3s, JPEGs, and so on). I've got an add-on USB-powered disk drive that probably wouldn't last more than a day or two of typical use on batteries. In any case, even if it ran on batteries, it would still need a cable (i.e., a wall wart or plug into a wired USB port) to charge them.

When it comes to wires, less is better than more, but it isn't the same as none. I guess that's why they don't call it "Wirefree USB." What the world needs now is a new-age Nikola Tesla to bring wireless power to the masses (see Figure 4).

Wireless USB will be a winner—but I wouldn't write off wires just yet.

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.

REFERENCE

[1] B. Hosler, "Certified Wireless USB Architecture Overview," 2006, www.usb.org/developers/wusb/docs/presentations/Taipei06_BH_Architecture.pdf.

RESOURCES

USB Implementers Forum, Inc., www.usb.org

UWB Forum, www.uwbforum.org

WiMedia Alliance, 2006, www.wimedia.org

SOURCES

CYRF6936 Radio module
Artaflex Modules, Inc.
www.artaflexmodules.com

CYRF6936 Radio SoC
Cypress Semiconductor Corp.
www.cypress.com

UWB Chipset and Certified Wireless USB modules
Staccato Communications, Inc.
www.staccatocommunications.com

IDEA BOX

THE DIRECTORY OF PRODUCTS AND SERVICES

AD FORMAT: Advertisers must furnish digital submission sheet and digital files that meet the specifications on the digital submission sheet. **ALL TEXT AND OTHER ELEMENTS MUST FIT WITHIN A 2″ × 3″ FORMAT**. Call for current rate and deadline information. Send your disk and digital submission sheet to: IDEA BOX, Circuit Cellar, 4 Park Street, Vernon, CT 06066 or e-mail adcopy@circuitcellar.com. For more information call Shannon Barraclough at (860) 872-3064.

The Suppliers Directory at www.circuitcellar.com/suppliers_dir/
is your guide to a variety of engineering products and services.

DSP workshops™

Embedded DSP Hands-On Workshops

Avoid confusing datasheets.

Rapidly learn to program, design and debug your application.

Reinforce course concepts with included development tools.

Let us help you accelerate your design process. Visit:
www.dsp-workshops.com

USB

Add USB to your next project—it's easier than you might think!

- USB-FIFO up to 8 mbps
- USB-UART up to 3 mbps
- USB/Microcontroller boards pre-programmed with firmware
- **2.4GHz ZigBee™ & 802.15.4**
- **RFID Reader/Writer**

Absolutely NO driver software development required!

www.dlpdesign.com

DLP Design

PHYTEC

phyCORE® OEMable Single Board Computers

XScale: PXA270, PXA255

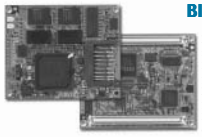
ARM: LPC3180 (ARM9); LPC22xx, LPC229x, AT91 (ARM7)

PowerPC: MPC5554, MPC5200B, MPC565, MPC555

ColdFire: MCF5485 **x86:** Elan SC520

C166/XC16x/ST10/8051 **CAN**

Blackfin: BF537



Faster-to-Market: Save time by integrating a PHYTEC Single Board Computer Module into your target circuitry.
Make -or - Buy: Why make your own when you can buy PHYTEC off-shelf solutions, cost-effective to 1000s units/year?

Integrated Support Services: Let PHYTEC assist you in the design of your end product: from tools and RTOSes to production. Our hardware is bundled with leading compilers (Keil, IAR, CodeWarrior), RTOSes (WinCE, Linux) and debuggers.
Immediate Support: Talk to PHYTEC technical staff with every call. No waiting for answers.
Your OEM solution: With 20 years design, production, and integration experience, PHYTEC is your OEM partner.

PHYTEC America, LLC ■ 203 Parfitt Way SW, G100 ■ Bainbridge Island, WA 98110 USA
www.phytec.com ■ (800) 278-9913 ■ www.phycore.com

TwinText

STRAIGHTFORWARD SOURCE CODE DOCUMENTATION

DOCUMENT ALL YOUR EMBEDDED CODE WITH A SINGLE TOOL

- C/C++, Assembler, BASIC, all their variants, plus 25 additional programming languages
- Easy to use: NO tags, XML, or specific comment boxes
- Automatic and scriptable code documentation and application help generation

Download your 30-day trial

Document From the Source

WWW.PTLOGICA.COM

GRAND VALLEY STATE UNIVERSITY
www.gvsu.edu

Faculty Position in

Electrical Engineering
or
Computer Engineering

Tenure-track assistant professor position available in embedded systems, VLSI/IC design, hardware/software co-design, and analog/RF hardware. GVSU has an ABET-accredited undergraduate engineering program and a practice-focused graduate program. A focus on student-centered quality teaching is foremost at GVSU.

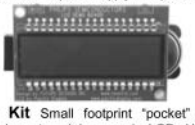
Apply on-line at:
http://www.gvsujobs.org

Grand Valley State University is an affirmative action, equal opportunity institution.

USB-ICP & LCD-Demo Kit



USB-ICP Using a standard USB port, USB-ICP supports In-System Programming (both ISP and ICP mode) for Philips 80C51, LPC9xx, and LPC2xxx ARM7 microcontroller families. In-System Programming uses a two-wire serial interface to program and erase ISP enabled microcontroller devices without removing them from the system. This device is powered over the USB connection, so no external power supply is required! **$69.00**.



LCD-Demo Kit Small footprint "pocket" LCD Demo board with 8-character alphanumeric LCD. Uses a highly integrated Flash based 80C51 device. A single CR2025 3V coin cell battery powers the entire board and the unit is User Re-Programmable via the ICP connector. **$49.00.**

For XA Development Kits and I2C, MDIO and SPI tools, please visit our website.

FDI
www.teamfdi.com
VISA/MC/Amex

Future Designs, Inc.
2702 Triana Blvd
Huntsville, AL 35805
(256) 883-1240
Fax (256) 883-1241

PROTOTYPE
CIRCUIT BOARDS

$85 US for two 5" x 6" 2-layer boards

Shipped NEXT BUSINESS DAY if data is received by 1:00 pm EASTERN

www.apcircuits.com

AP Circuits

(403) 250-3406

staff@apcircuits.com

MOTION CONTROL

Real Time Motion Control Design.
Custom Firmware Design
Device Driver Layer Design
Real Time Operative System
Embeded TCP/IP Stack, USB 2.0
Zigbee, 802.11x, Bluetooth
Embedded Multi-Axes Motor Control
Data Logger Design ,*Home Automation*



SPORT232ST

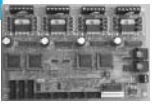
STEPPING CONTROLLER

Control up to 16 STEPPER MOTORS simultaneously using a "SINGLE SERIAL PORT".

Up to 8500 Steps/sec with linear Accel/Decel.
Chopper driver up to 48V/2A for 4,5or 6 wire motors.
Software Winding Current Setup.Shaft encoder input
Internal FIFO for linear, Linear or Spline Interpolation.
24 digital I/O.Sequential Motion Pivot Controller .
32 bit DLL **Windows 95,98,00,NT,Vista; Linux, LabView** *FREE SOURCE CODE* in C++,VBASIC

Embedded Web-Motion

SPORT232-STI
Stand Allone One Axis Motion System.
ST400NT-S



RMV Electronics Inc.
300-3665 Kingsway
Vancouver, BC, Canada V5R 5W2
Tel: (604) 299-5173 Fax: (604) 299-5174
Web Site : www.rmv.com

Advertising Calendar

June, Issue #203
Communications
Space Close: Apr. 12

BONUS DISTRIBUTIONS:
NXTcomm; RoboBusiness

July Issue #204
Internet and Connectivity
Space Close: May 11

Call: Shannon Barraclough
(860) 872-3064
Shannon@circuitcellar.com



$189

Tiny Linux controller

16MB fast SDRAM
64MHz Coldfire MCU
4.5 MB flash memory
SD card socket (to 2GB)

10/100 Ethernet
3 serial ports (RS232)
1 CAN port
LCD/KPD port

Eclipse/CDT dev. env.
Free serial debugger

55 I/O pins & capabilities to burn...

www.steroidmicros.com

WIRELESS RS-232

WCSC (Willies Computer Software Co)



- Extremely easy to install and use
- Distances up to 100 meters (329 feet)
- Bluetooth Serial Port Profile
- Communicates with Bluetooth enabled PDAs, Laptops, Smartphones, & Computers
- No programming or special software needed

Other Products

- Professional serial communication libraries & development tools.
- PCI, PCMCIA, USB, Universal PCI, & ISA multiport RS232, Rs422, & Rs485 cards, Bluetooth/USB Dongles

http://CircuitCellar.wcscnet.com
sales@wcscnet.com (281)360-4232

Low Cost EDA Software For Windows 98, XP, NT, 2000

- Circuit design package with schematic entry, circuit board layout and simulation (SPICE and digital) for only $599
- Buy modules for as low as $119
- Purchase and download products instantly from our website!
- Libraries include latest micros, memories, analog, and others

Contact **Mental Automation** at
(253) 858-8104
http://www.mentala.com

High-speed USB 2.0 + Xilinx FPGA Complete Software API



75mm x 50mm
$399.95



3.5" x 2.0"
$199.95

XEM3010-1500P:
- 1,500,000-gate FPGA
- Programmable PLL
- Over 110 I/Os
- 32 MB SDRAM
- Configuration PROM
- 0.8-mm expansion
- $399.95 / Qty 1
- $349.95 / Qty 10

XEM3001:
- 400,000-gate FPGA
- Programmable PLL
- Over 80 I/Os
- 0.1" expansion headers
- Business-card size
- $199.95 / Qty 1
- $174.95 / Qty 10

FrontPanel Software API:
- Windows XP, Mac OS X, Linux
- C/C++, Python, Java
- Configure FPGA *and communicate* with your design
- The easiest way to integrate USB into your product
- Use for image capture, control, test equipment, etc.
- Up to 38 MB/s transfer rate!

*5% off with Coupon Code: CKTCLR73**

* Valid for first order only. Void 30-days after issue publication.

Opal Kelly
Visit us online at:
www.opalkelly.com

Start designing with Freescale's advanced S12X family!

Technological Arts

NEW!

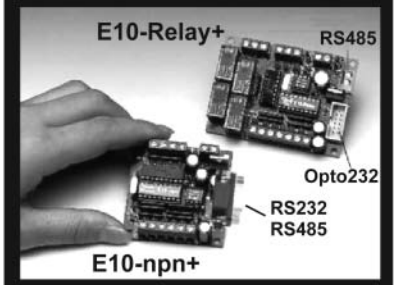


freescale Alliance Member

Toll-free: 1-877-963-8996
www.technologicalarts.com

The $69 PLC

Work as Stand-Alone Ladder Logic PLC.
Or as Smart Remote I/Os of PC/ PLCs.
RS485 allows 256 units to be networked.



Incredibly Easy to Program!
Our software is used by many colleges for teaching PLCs!

Get Free Ladder Logic Simulator:
www.tri-plc.com/cci.htm

TRI Triangle Research

Tel: 1-877-874-7527 - PLC specialist since 1993

www.can232.com

Only $108 €89

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 independant
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

The original (Controller Area Network) CAN to RS232 serial converter.

Only $154 €129

www.canusb.com

Instant LCD



ezLCD-002 - $189

- Versatile Programmable Module
- Serial/I²C/USB/Parallel Interface
- AVR/BASIC Stamp/VB Compatible
- Onboard Flash Bitmap Memory
- Downloadable TTF Fonts
- 2.7" or 5.6" TOUCH Color TFT LCD
- 240x160 or 320x240 resolutions
- Transflective w/ LED Frontlight (2.7")
- Transmissive w/ 350 nit backlight (5.6")
- 512 colors or 65,535 colors

EARTHLCD.com We Make LCDs Work.™

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



LEMOS INTERNATIONAL

Call Toll Free 1-866-345-3667 or email at sales@lemosint.com

www.lemosint.com

PC/104 PERIPHERALS

SCIDYNE Offers a Full Line of Innovative Modules for PC/104 Applications

ADIO-104 This Module Does it all!
- 16 Analog Inputs
- 8 AnalogOutputs
- 24 Digital I/O lines
- 5V Power
- Pulse Accumulator
- Open-Drain Outputs

DIO96-104
- 96 Bi-Directional Digital Channels
- Input, Output and Strobed I/O functions
- Uses Familiar 82C55A Chips

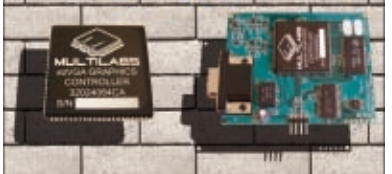
XIO-RO8

Add eight High-Power relay outputs to any digital port.

Call or Visit us on the web

SCIDYNE® 1-877-724-3963 www.scidyne.com

Color VGA video doesn't have to be difficult...



Introducing the new embedded ezVGA Graphics Controller chip and ezVGA Serial Video Module. Color VGA graphics has never been so easy!

For more product information:
Web: www.multilabs.net
Email: support@multilabs.net
Phone: (949) 458-7625

MULTILABS



RabbitFLEX
A New Way to Customize
- Click-to-ship in 5 days
- Pay only for what you need
- No NRE Charges
- Perfect for prototype or production

Kits From $399

RABBIT Semiconductor

www.RabbitFLEX.com

Solve complex signal acquisition problems...
- positioning & control
- environmental
- acceleration
- transients
- pressure
- vibration
- sonar
- GPS



- Linux Driver
- Guaranteed in stock
- Customization available
- 16-bit analog inputs and outputs
- Million sample FIFO eliminates interrupts
- Wide analog input and output ranges
- -40°C to +85°C Standard

www.stx104.com
Apex Embedded Systems
sales@stx104.com • 608-256-0767 x24



GOT IR?

FOR DATA OR CONTROL -
An IR Buddy can help!

A WIZARD at sending and receiving IR data, control codes and detection beams, it can link two BASIC Stamps or Boe-Bots up to 50 feet apart, detect objects and people, be a stand-alone IR repeater, and more. Small size (0.875" by 0.5") and low power add versatility. On-chip 8-byte data-buffer eases Stamp programming.

Formerly by Parallax Inc., the IR Buddy is now available as an Accessory for the BASIC Stamp

Get IR! Only online at: www.irbuddystore.com

I²C™/SMBus
- Bus Monitors
- Protocol Analyzers
- Host Adapters
- Multiplexers
- Battery Applications
- Software Tools



MCC Micro Computer Control

I²C is a trademark of Philips Corporation

www.mcc-us.com

New IP Products • $129.95



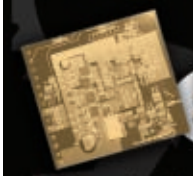
IP Power
- 4 independent power outlets
- SDK for complete customization



IP Video
- NEW SDK tools for customization including telnet access, busybox utilities, wget, and more
- New simultaneous 4 channel DVR compatibility

Aruca electronics
www.ArucaElectronics.com

Electrical Design
Mechanical Design
Software, PCB Layout
Prototyping Services



Quick-turn PCB Prototype Assembly As few as one board Call for a free quote

Engineering Your Path to Profit

IMET Corporation
2014 Ford Road, Unit B • Bristol, PA 19007
Tel: 215-788-5256 • 866-435-4907
Fax: 215-788-5362
www.imetcorporation.com

Low Cost CAN USB Adapter

Simple configuration & use

$99.95 Qty 1



+1 630-245-1445
Naperville, Illinois USA
www.c-a-n.com
gridconnect

USB Device Stack

- Compatible with Windows USB 2.0 Host Class Drivers:
  - Mass Storage
  - Mouse
  - Ethernet over USB
  - Serial
- Composite Devices
- ARM, ColdFire, Philips ISP 1161, 1181, 1362, 158x, 1761
- Low Cost, Royalty-Free
- Full Source Code
- Standalone or RTOS
- Host and OTG Available

Micro Digital Inc
RTOS Innovators
www.smxrtos.com/usb

1,600+ Products
www.DesignNotes.com





DesignNotes.com
Affordable Prototyping Products
1-800-957-6867

ValueCAN

The High Value Tool For Controller Area Network

- USB to CAN
- Simple software analyzer included
- DLL with examples for custom applications
- PC isolated from CAN
- 100% bandwidth at 500Kb

Only $295

ICS RADDAQ
www.intrepidcs.com

CONTROLLER AREA NETWORK
CAN to RS232 CAN to USB

Flash CAN232 - The original RS232 to CAN adapter. Supports both ASCII and binary commands. *$99*
CANPIC CAN232 – Enhanced CAN to RS232 adapter just *$89*
***NEW!* CANUSB -** CAN to USB adapter *$99*
***Lowest!* TrashCAN -** Bare bones CAN to USB adapter as low as $69.95.
I2C232 - I2C to RS232 adapter only *$89*.
I2CUSB - I2C to USB adapter only *$89*.

Need a Software Guy?

Over 25 years experience from high volume automotive components to one-of-a-kind test boxes. Let EMICROS do your next embedded software project.

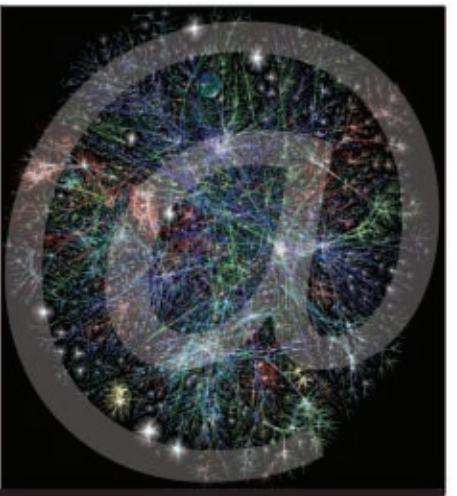
EMBEDDED MICRO SOFTWARE
email: info@emicros.com

www.EMICROS.com
sponsored by ThePokerBug.com

CIRCUIT CELLAR
Designer's Notification Network

Circuit Cellar design contest entrants have received thousands of valuable development tools and product samples. Because of their contest participation, these engineers receive advance e-mail notice from Circuit Cellar as soon as new samples become available. Now you too can benefit from this early notification.

Welcome to the Designer's Notification Network. Print subscribers are invited to join the Network for advance notice about our new sample distribution programs.



Find out more at www.circuitcellar.com/network.

CHINA PCB SUPPLIER

DIRECT SALE FROM CHINA
PROTOTYPE TO PRODUCTION

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



WWW.PCBCART.COM

MSP430
CPU12
ARM
AVR

Easy to Use! Low Cost! State of the Art!

IMAGEcraft C compilers

starting at $199!

high powered development, garage powered prices!



Fully Functional Demos!

Plus:
AVR, ARM & HCS12 Hardware Kits

www.imagecraft.com
info@imagecraft.com
(650) 493-9326 • FAX (650) 493-9329

Link Instruments
Digital Oscilloscope
500 MSa/s

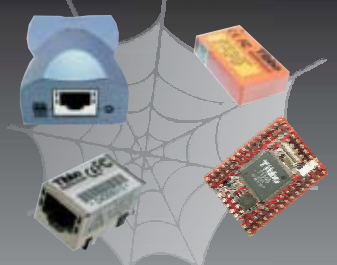
- 2 Channel Digital Oscilloscope
- 500 MSa/s max single shot rate
  250 MSa/S (Dual channel) 512 Kpts
  500 MSa/S (Single channel) 1 Mpts
- Advanced Triggering
- Portable and Battery powered
- Only 9 oz and 7" x 3.5" x 1.5"
- FFT Spectrum Analyzer
- USB 2.0

$950



www.LinkInstruments.com 973-808-8990

Net Modules



programmable revolutionary

SAVE 5% enter coupon code CCI at checkout
ABACOMdirect.com
ABACOM Technologies
tel: +1(416) 236 3858
fax: +1(416) 236 8866

Zigbee
watchout
here comes
Spider
wireless
auto-binding multidrop sensor / actuator network pods

SAVE 5% enter coupon code CCI at checkout
ABACOMdirect.com
ABACOM Technologies
tel: +1(416) 236 3858
fax: +1(416) 236 8866

Wireless I/O

extend your digital I/O's over 100's of feet up to tens of miles - in both directions - with the 16IO-SSRT RF transceiver modules



SAVE 5% enter coupon code CCI at checkout
ABACOMdirect.com
ABACOM Technologies
tel: +1(416) 236 3858
fax: +1(416) 236 8866

MYLYDIA, INC.

Layout Gerber,
Prototype Making

QUICK TURN

PCB & Turnkey
@
The Best Prices

Tel: 800-695-9342
Sales@mylydia.com
WWW.MYLYDIA.COM

RD™ Low Cost, High Performance, Versatile Controller

80 MHz

NEW



- 3.9 x 3.6" C/C++
- 20 Solenoid Drivers
- Ethernet, 3 RS232, 16 ADCs, 8 DACs and 40+ IO
- 1 GB CF mass data storage with file systems
- 80 MHz CPU, 16-bit Flash and SRAM

OEM $99

40+ Low Cost Controllers with ADC, DAC, solenoid drivers, relays, file system with CompactFlash, LCD, DSP motion control, 11 UARTs, 300 I/Os. 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

CAN-4-USB FX

USB to CAN Interface
USB 2.0 Hi-Speed 480Mbps!

Other companies may claim USB 2.0 but if it isn't Hi-Speed it is only 12Mbps.

2007 marks our 11th year of selling CAN interfaces in over 30 countries!



$249 USD.

Zanthic Technologies Inc.
403-878-2202
www.zanthic.com

8/16-bit EEPROM | Ser. EEPROM | Flash EPROM | GAL / PALCE | Most MCU's | Low Voltage to 1.3V. | DIL dev. w/o Adapter.

Conitec's last generation Galep-4 employs ASIC universal pin technology for each pin of 40 pin ZIF-socket. **10,000+device library** lifetime free updates. Handle 8/16 bit EPROM'S, EEPROM's, 0-Pwr RAM, FLASH, Serial EEPROM's, GAL, PALCE, microcontrollers such as 87/89xxx, PIC, AVR, ST62, etc. Low voltage devices down to 1.3V. No adapter required for DIL devices. 8 Hrs. operation on battery (AC charger included). Runs **WIN 98,NT,ME, 2000, XP** with Hex/Fuse Editor. Remote control from other apps, (e.g. VisualBasic). Substitutes high priced universal programmers e.g. ALL-11 (HILO) or LAB-TOOL-48 (ADVANTECH) **Provides virtually matching performance at 1/3 to 1/5 the price.** Info, orders, softwr : **619-462-0515**

ONE PALM-SIZED PROGRAMMER HANDLES OVER 10,000 DEVICES ?

ISO 9001:2000 CERTIFIED PROFESSIONAL TOOLS

9001: 2000

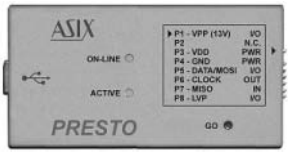
10,000 DEVICES



TINY PACKAGE. 10,000 DEVICE OUTPUT.

SALES, SUPPORT / EMAIL: CONTACT@CONITEC.NET / WWW.CONITEC.NET TEL: 619-462-0515. FAX: 619-462-0519

PRESTO In-System USB Programmer



Very fast and flexible in-system programmer for Microchip & Atmel MCUs, serial EEPROMs & Flash PROMs, devices with JTAG (CPLDs, MCUs, FPGAs, FPGA PROMs, ...)

SIGMA - unique logic analyzer
USB modules - parallel & serial

ASIX **ASIX**, Prague, Czech Rep.
info@asix-tools.com

www.asix-tools.com

SpectraPLUS 5.0
Audio Spectrum Analysis

Features
Sound Card based I/O
FFT sizes to 1048576pts, 1/96 Octave
Up to 24 bit, 200kHz sampling rates
3-D Surface and Spectrogram
Digital Filtering, Signal Generation
THD, IMD, SNR, Transfer Functions
DDE, Macros, Data Logging,
Vibration Analysis, Acoustic Tools

FREE 30 day trial!
www.spectraplus.com

PHS Pioneer Hill Software
360 697-3472 voice
pioneer@telebyte.com

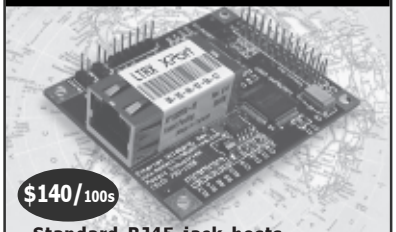
Discover MCU Space, a comprehensive place of PIC® MCU tools, articles, and links.

visit www.MCUspace.com



PIC is a registered trademark of Microchip Technology Inc. in the U.S. and other countries.

EtherSmart Wildcard™
Network -Enables Your Product



$140/100s

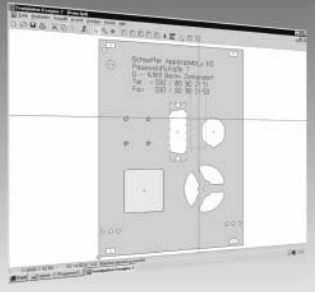
- Standard RJ45 jack hosts 10/100Mbit Ethernet
- HTTP, SMTP, TCP, DHCP, ICMP, and ARP Protocols
- Email program-controlled messages to a specified LAN IP address
- Establish a TCP/IP connection to exchange binary or ASCII data
- Serve software-controlled dynamic content to your web browser

Mosaic Industries Inc.
tel: 510-790-1255 fax: 510-790-0925
www.mosaic-industries.com

Front Panels?

Unrivaled in price and quality for small orders

Download the free ›Front Panel Designer‹ to design your front panels in minutes





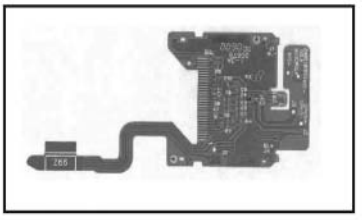
Order your front panels online and receive them just in time

www.frontpanelexpress.com

Serial • Ethernet
Web Server
Simple, DTE, DCE
No SW Changes
$99.95 Qty 1
+1 630-245-1445
Naperville, Illinois USA
www.gridconnect.com
gridconnect


ANYONE
Can Now Easily
Hand Solder Surface-
Mount Components!
Even A 10
Year Old!
www.schmartboard.com


Weather
Instruments
for PCs
AAG electrónica
www.aagelectronica.com

High Quality FPC
Competitive Price

Start at
$299
Qty10
WWW. EzPCB. COM

Embedded Ethernet
ONLY $98

NEW! picoFlash
• 10Base-T Ethernet
• 186 Processor @ 40 MHz
• DOS w/ Flash File System
• 16 Digital I/O • 5V DC
• Console / Debug Port
• Hardware Clock/Calendar
• Socket for DiskOnChip
• 512K DRAM & 512K Flash
• (2) Serial Ports
• (2) 16-bit Timers
• Watchdog Timer
• 3.75" x 2.50"
$129 Development System
Development Kit Includes:
• picoFlash Controller
• Borland C/C++ 4.52
• TCP/IP, PPP & Web Server
• Serial Driver Library
• AC Adapter and Cables
Call 530-297-6073 Email sales@jkmicro.com
On the web at www.jkmicro.com
JK microsystems

USB
STEP MOTOR CONTROL
HS-20USB
• 4 axis control w/ external step & direction / full or half step drives
• encoder support
• digital I/O
• USB 2.0
• qty 100 $499 ea

CyberPak COMPANY INC.
800-328-3938 • www.cyberpakco.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
Free 96 page catalog
1-800-826-5432

PRINTED CIRCUIT BOARDS
QUALITY PRODUCT
FAST DELIVERY
COMPETITIVE PRICING
PROTOTYPE through PRODUCTION
• Aluminum Backed PCB
• Single & Double sided
• SMOBC/RoHS
• LPI mask
• Through hole or SMT
• Nickel & Gold Plating
• Routing or scoring
• Electrical Testing
• Artwork or CAD data
• Fast Quotes
• Flex Circuits
SPECIAL OFFER:
10 pcs (3days) 1 or 2 layers $249
10 pcs (5days) 4 layers $695
(up to 30sq. in.ea.) includes tooling, artwork, L.P.I. mask & legend
PULSAR, INC
9901 W. Pacific Ave. Franklin Park, IL 60131 • Phone 847.233.0012
Fax 847.233.0013 • www.pulsar-inc.com • sales@pulsar-inc.com

Full Speed
It writes your USB Code!

NEW! HIDmaker FS for Full Speed FLASH PIC18F4550

Creates complete PC and Peripheral programs that talk to each other over USB. Ready to compile and run!

- Large data Reports
- 64,000 bytes/sec per Interface
- Easily creates devices with multiple Interfaces, even multiple Identities!
- Automatically does MULTITASKING
- Makes standard or special USB HID devices

NEW! "Developers Guide for USB HID Peripherals" shows you how to make devices for special requirements.

Both PC and Peripheral programs understand your data items (even odd sized ones), and give you convenient variables to handle them.

PIC18F Compilers: PICBASIC Pro, MPASM, C18, Hi-Tech C.

PIC16C Compilers: PICBASIC Pro, MPASM, Hi-Tech C, CCS C.

PC Compilers: Delphi, C++ Builder, Visual Basic 6.

HIDmaker FS Combo: Only $599.95

DOWNLOAD the HIDmaker FS Test Drive today!

Trace SYSTEMS, Inc.

www.TraceSystemsInc.com

301-262-0300

Controllers

NEW ET-AVR Stamp Module



- Includes ATMega64 Microcontroller
- 64kbytes Flash Program Memory
- Up to 53 I/O Points
- 8 Channel 10-bit A/D
- Ideal As plug-in Controller
- Direct Program Download

Also Available: ET-AVR Stamp Development Board with P/B's, LED's, LCD Connector etc

www.futurlec.com

USB Data Acquisition

ADU208 -USB Relay I/O Interface



FEATURES
-8, 5-AMP relay outputs
-8, ISOLATED digital inputs
-Port Powered, Aux 5VDC output $189.00 QTY 1

Other Models:
ADU200- 4 Channel Version with RS232 $139.00
ADR218- Solid-State Version 8 Channel $225.00
ADU100- 3 CH, 16-Bit ISOLATED Analog Inputs, PGA, 4 digital I/O, RS232 and 5 AMP Relay Output $199.00

ONTRAK CONTROL SYSTEMS INC.
PH: (705) 671-2652 Fax: (705) 671-6127

www.ontrak.net



JStamp!®

FAST 32-bit native-execution Java
Executes 3 million Java byte codes/sec
compact 1 x 2 inch DIP40 package
Real J2ME CLDC Java (threads, floats)
2048 KB Flash & 512 KB SRAM

www.jstamp.com

SYSTRONIX
Salt Lake City, Utah, USA

Java is a TM of Sun Microsystems, Inc. JStamp is a TM of Systronix, Inc

NAND FLASH FILE SYSTEM

- Wear Leveling, Garbage Collection
- Power-Fail Safe
- ECC and Bad Block Handling
- File System with Std C Lib API
- FAT File Structure
- Page Caching to Maximize Performance and Minimize RAM
- 20KB Code Size
- RAM Required < 1% Flash Size

www.smxrtos.com/ffscc.htm

Micro Digital Inc
RTOS Innovators

800.366.2491 sales@smxinfo.com

Order online at: www.melabs.com

microEngineering Labs, Inc.

Development Tools for PIC® Microcontrollers

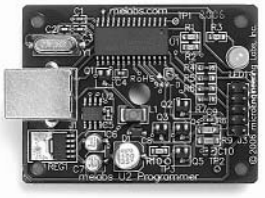
Phone: (719) 520-5323
Fax: (719) 520-1867
Box 60039
Colorado Springs, CO 80960

USB Programmer for PIC® MCUs — $89.95 (as shown)

RoHS Compliant

Programs PIC MCUs including low-voltage (3.3V) devices

Includes Windows 98, Me, NT, 2K, and XP Software



With Accessories for $119.95: Includes Programmer, Software, USB Cable, and Programming Adapter for 8 to 40-pin DIP.



EPIC™

Parallel Port Programmer starting at $59.95

Serial Port Programmer starting at $79.95

LAB-X Experimenter Boards



Pre-Assembled Boards Available for 8, 14, 18, 28, and 40-pin PIC® MCUs
2-line, 20-char LCD Module
9-pin Serial Port
Sample Programs
Full Schematic Diagram

Pricing from $79.95 to $349.95

PICPROTO™ Prototyping Boards



Double-Sided with Plate-Thru Holes
Circuitry for Power Supply and Clock
Large Prototype Area
Boards Available for Most PIC® MCUs
Documentation and Schematic

Pricing from $8.95 to $19.95

BASIC Compilers for PICmicro®



Easy-To-Use BASIC Commands
Windows 9x/Me/2K/XP Interface

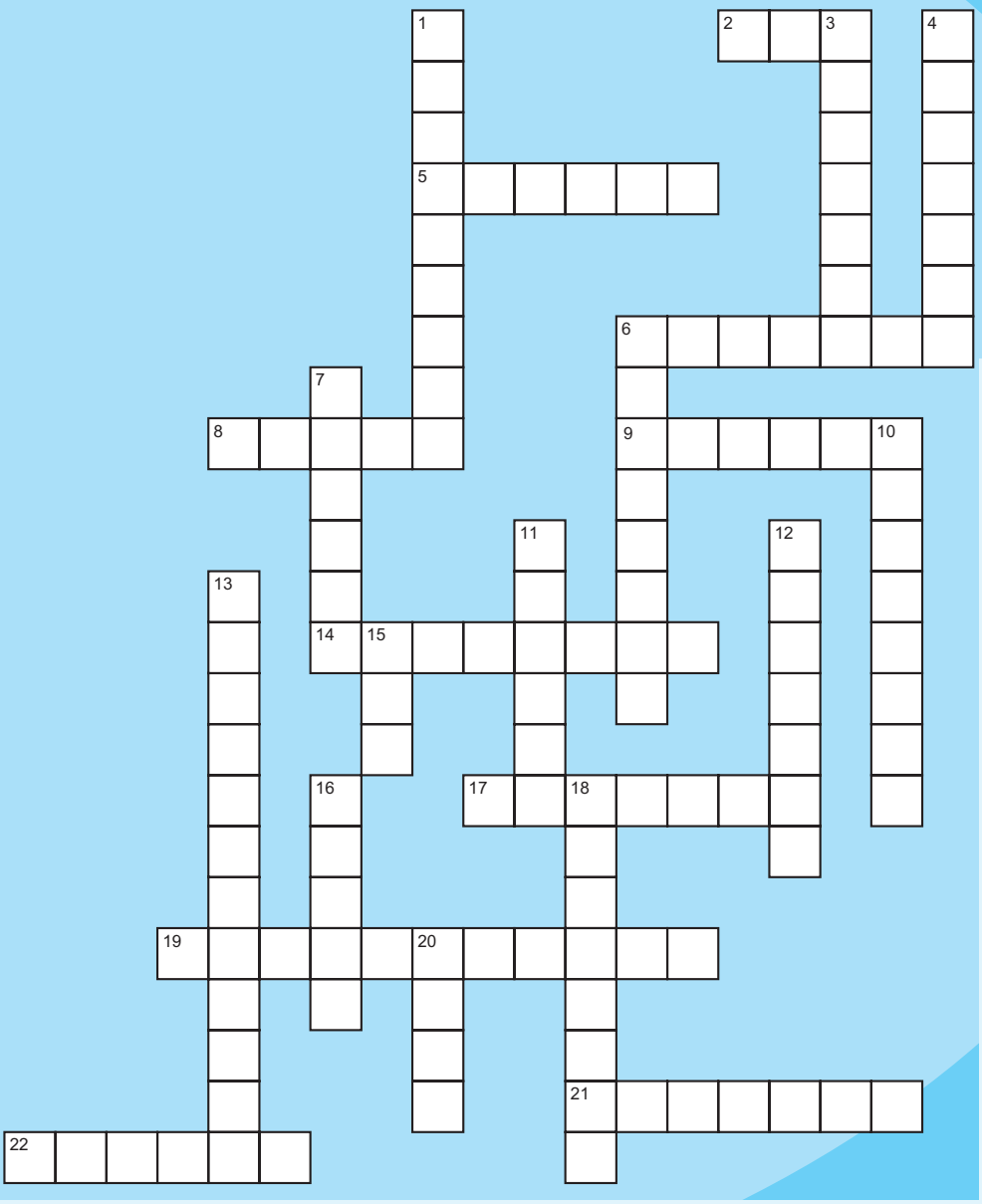
PICBASIC™ Compiler $99.95
BASIC Stamp 1 Compatible
Supports most 14-bit Core PICs
Built-In Serial Comm Commands

PICBASIC PRO™ Compiler $249.95
Supports 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

CROSSWORD



Across

2. An item used to remotely open a car or identify a key ring
5. Erasable memory
6. Surrounding; enclosing
8. A self-service public computer system that is set up for information browsing
9. An open-source software package used as a digital repository
14. An array of frequencies admitted or absorbed by a substance
17. A program used to access the Internet
19. A signal that must be delivered within a specific set of equal time limits
21. A declaration in C language to give a variable type a new name and make the program more readable
22. A type of metal transistor

Down

1. A word or phrase that links to another document when clicked with a mouse
3. The British mathematician (1791–1871) who designed the first digital computer
4. A signal sent after a program has waited a specified length of time for an event that did not occur
6. An additional item added to the end of a datasheet
7. A communications protocol that allows Modicon-programmable controllers to communicate with each other
10. The type of programming featured in *Circuit Cellar* this month
11. Noise or incorrect signals added to a digital signal to reduce the amount of digital distortion
12. A basin-shaped volcanic crater
13. A device that displays various electronic quantities on a screen in the form of voltages
15. File extension for Photoshop files
16. Supplemental software that you add to a program to fix problems
18. What is outdated hardware or software?
20. A break in a planet's surface

The answers are available at www.circuitcellar.com/crossword.

Brought to you by:

LEMOS INTERNATIONAL

INDEX OF ADVERTISERS

The Index of Advertisers with links to their web sites is located at www.circuitcellar.com under the current issue.

Page		Page		Page		Page	
91	AAG Electronica, LLC	85	FDI-Future Designs, Inc.	23	Maxstream	19	Pololu Corp.
86	AP Circuits	90	Front Panel Express, LLC	86	Mental Automation	91	Pulsar, Inc.
90	ASIX	92	Futurlec	87	Merican Made	86	RMV
89	Abacom Technologies	89	General Circuits, Inc.	88, 92	Micro Digital, Inc.	73	R4 Systems
91	All Electronics Corp.	85	Grand Valley State University	63	Microchip	34, 64	Rabbit Semiconductor
87	Apex Embedded Systems	88, 91	Grid Connect	15	Microchip 16-bit Embedded Control Design Contest	87	Rabbit Semiconductor
24	Arcom Control Systems	66	HI-TECH Software LLC			22	Reach Technology, Inc.
88	Aruca Technologies	55	Holmate/Holtek Semiconductor, Inc.	92	microEngineering Labs, Inc.	9	Renesas
7	Atmel	89	IMAGEcraft	90	Mosaic Industries, Inc.	81	Saelig
42	Bitscope Designs	86	Intec Automation, Inc.	95	Mouser Electronics	91	Schmartboard
65	CWAV	88	Intrepid Control Systems	87	Multilabs	87	Scidyne
80	CadSoft Computer, Inc.	11	Ironwood Electronics	89	Mylydia, Inc.	26	Sealevel Systems
90	Conitec	64, 91	JK microsystems, Inc.	5	NKK Switches	82	Sensors Expo & Conf.
90	Custom Computer Services, Inc.	50	Jameco	C2	NetBurner	17	Silicon Laboratories, Inc.
91	CyberPak Company, Inc.	54	Jeffrey Kerr, LLC	3	Noritake Co., Inc.	92	Systronix
1	Cypress	74	Keil Software	52	Nurve Networks LLC	C3	Tech Tools
85	DLP Design	52	LabJack Corp.	92	Ontrak Control Systems	48, 49	Technologic Systems
85	DSP Workshops	52	Lakeview Research	86	Opal Kelly Inc.	86	Technological Arts
54	Decade Engineering	31	Lantronix	47	P&E Micro	89	Tern, Inc.
88	Designnotes.com	87	Lawicel AB	10	PCB-Pool	58	Tibbo Technology, Inc.
33	EMAC, Inc.	11, 87	Lemos International	85	PTLogica	92	Trace Systems, Inc.
87	Earth Computer Technologies	2, 89	Link Instruments	C4	Parallax, Inc.	40	Tri-M Systems, Inc.
88	eMicros	33	Linx Technologies	85	Phytec America LLC	86	Triangle Reasearch Int'l, Inc.
30	ExpressPCB	87	MCC (Micro Computer Control)	27	Pico Technology, Ltd. USA	86	WCSC (Willies Computer Software Co.)
25, 91	ezPCB	41	Matrix Orbital	90	Pioneer Hill Software	90	Zanthic Technologies, Inc.

Preview of May Issue 202
Theme: Measurement & Sensors

Proton Precession Magnetometer

Energy and Load Analyzer

Being an Embedded USB Host

Circuit Board Plotting

Temperature Calibration System

The Witness Camera: Build a Self-Recording Surveillance Camera

---

LESSONS FROM THE TRENCHES **More "Hello World":** C Program Flow

FROM THE BENCH **What's the Slope?: Using an Accelerometer to Measure Slope**

SILICON UPDATE **Be All You Can Bee**

ATTENTION ADVERTISERS

June Issue 203 Deadlines

Space Close: April 12
Material Close: April 20

Theme: Communications

BONUS DISTRIBUTION:
NXTcomm, RoboBusiness

Call Shannon Barraclough now to reserve your space!
860.872.3064
e-mail: shannon@circuitcellar.com

XBee™ ZigBee OEM RF Module



ZigBee/IEEE 802.15.4 compliant solution that delivers critical data between devices. Operates within the ISM 2.4GHz-frequency band, 1mW power output up to 100m range, U.FL.RF connector, -40°C to 85°C temp. range, and chip or wire options. Supports advanced networking and low-power modes.

MaxStream® mouser.com/maxstream/a

XPort™ AR



XPort™ AR contains all the essential device computing and networking features built into a single RJ45 package, including a 120Mhz DSTni™ processor, 10/100 Base-T/TX Ethernet connection, an embedded web server, and a full network operating system with TCP/IP protocol stack.

LANTRONIX® mouser.com/lantronix/a

Embedded Products for the Latest Technologies



Rabbit Semiconductor BL2600 Wolf™

Reduced time to market is critical for new product designs -- lost time means lost revenue. That's why engineers depend on Mouser to deliver a broad selection of embedded products fast!

And because these components have a solution designed in, engineers can utilize these plug-and-play modules from test through production - saving time and money.

Experience Mouser's time-to-market advantage! Our vast selection of the NEWEST products, NEWEST technologies, **new catalog every 90 days,** no minimums, and same-day shipping on most orders, gets you to market faster. We make it easy to do business with Mouser!

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

MOUSER ELECTRONICS®
a tti company

The Newest Products For Your Newest Designs

FlashDisk Module



Provides non-volatile, solid state data and program storage for embedded applications. Operating system, application software compatibility, and portability is ensured by the module's True IDE interface. No moving parts allow for more rugged and reliable performance than rotation hard drives.

STEC mouser.com/stec/a

4000 High-Performance Microprocessor



Low EMI microprocessor for embedded control, communications, and Ethernet connectivity. Glueless architecture, 10Base-T Ethernet, C-friendly instruction set. Up to 60MHz, 8 independent DMA channels, supports 8 or 16-bit Flash and SRAM memories, 7 hardware breakpoints, 40+ I/O lines.

RABBIT Semiconductor mouser.com/rabbitsemi/a

MOD5282 32-bit Processor Module

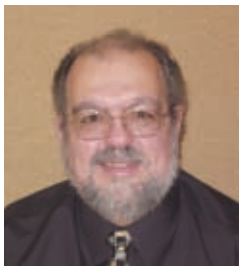


32-bit 66MHz ColdFire processor with 10/100 Ethernet, 3 UARTs, 8 10-bit ADC, I2C, SPI, and 50 digital I/O. Comprehensive software package consists of uC/OS RTOS, TCP/IP stack, web server, NetBurner Eclipse IDE with C/C++ compiler, and debugger.

NetBurner Networking in 1 Day! mouser.com/netburner/a

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

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



PRIORITY INTERRUPT

by Steve Ciarcia, Founder and Editorial Director

Only Real Events Lead to Real Trends

There have been lots of blogs and online articles stipulating that print media is dead, but let's face it, they are mostly online publishers and have no particular reason to say otherwise. I'm not swayed by one-sided dialog. Only real events lead to real trends.

Two recent events in particular caught my attention, both at *PC Magazine*. In the February 6 issue, columnist Michael Miller announced that his column would no longer be in print, but that he would still write a "columnist's blog" on the magazine's web site. Soon after that, Bill Machrone announced in his column that he too was stepping down. To my knowledge, both of these guys are former editors-in-chief of *PC Magazine* and should have influence where it counts. This hardly jibes with Bill's comment that, "As the technical publishing industry continues trying to find the balance between print and online media, economic forces weigh against the column's continuance." ("Wrapping Up," *PC Magazine*, January, 31, 2007) Admittedly, he wasn't saying print is dead, but it sure leaves a lot of speculation about what he and Michael will really be doing for ZD in the future.

These guys aren't the only ones. An editorial by Josh Norem at www.hardocp.com is about how he, the former editor-in-chief of *Maximum PC* print magazine, is now at *HardOCP* online magazine because he thinks print magazines like *Maximum PC* are dying: "These days, readers flock to their favorite website, and instead of gauging public interest in terms of the number of issues sold, we measure it in page views." ("Magazines vs. the Web: End of an Era," HardOCP, February 5, 2007) Of course, there is also this little thing about being priced out of the housing market where he lived when he worked for *Maximum PC* that influenced his decision, but heh….

Seriously, if I tone down his rhetoric a bit, I might agree with Josh to some extent. Magazines that are entirely dependent on breaking news or up-to-the-minute product reviews in a fast-selling market are in a world of hurt. Print magazines have a lead-time associated with physically printing and distributing paper that adds up to weeks. Online magazines and news sites can post things online directly from an editor's desktop. The criticism is that why should you pay for "old" content in a print magazine when you can get it free and quicker online.

The car was supposed to eliminate the horse, television was supposed to eliminate radio, radio was supposed to eliminate newspapers, e-mail is supposed to eliminate the post office, etc, etc. Give me a break. Technology is evolutionary (and, BTW, I heard a recent statistic that there are more horses in the U.S. today than there were in 1900). Just because one segment of an industry is discovering intense competition, it doesn't mean the whole industry is suddenly dead. It just means that the concept of publishing has hybridized to include a range of being completely online at one end and all paper and ink at the opposite end.

I like to think that *Circuit Cellar* is situated correctly on the curve, but it's always a balancing act to know how far and how fast to move along it. Print magazines used to simply rely on the convenience of paper as their greatest raison d'etre—high-resolution photos, high physical content, and little concern about losing paper. This was compared to reading relatively low-resolution, low-capacity content stored on a bulky and expensive laptop that you definitely didn't want to leave on the train. Today, the same is not true. Even though displays don't yet reach *Circuit Cellar*'s 2,550 × 3,300 paper page resolution, a cheap super-high resolution OLED or an e-paper with an integrated wireless connection to an infinite amount of downloadable media is only a few development generations away. When that happens, it will just be defined as the new print media for us. *Circuit Cellar* intends to be right there on that OLED too.

The point that's missing in everyone's dire predictions is that they are confusing the words "print media" with the problems arising from "traditional concepts about print media content." Online or paper, magazines survive only if people value their content. In a situation when the source information is presented in parallel to all media at the same time, such as news events and new products for review, the publisher who gets it to readers first is the clear winner. The only mitigating factor that eliminates publication timing as an overriding issue is where the content is entirely unique or there is significant value added to the content by the publisher. In the case of *PC Magazine*, its future relies on maintaining the quality of its product reviews, not just doing them quicker.

In the case of *Circuit Cellar*, we like to think that it is our unique high-quality content that keeps both readers and advertisers coming back for more. Yes, *Circuit Cellar* is a print magazine today, but it's also part of an evolution that is always transforming. A dozen years ago we had no web site and were strictly print. Today, we are both an online and a print magazine with a very significant web presence.

And, just so you know that I follow my own advice, our hybrid methodology endeavors to dispel the greatest limitation of these traditional print media publishing concepts: content size. I did a little quick math before writing this editorial and it's an interesting statistic. As a print magazine, we publish approximately 750 pages of design articles each year in the print magazine. However, if you want to experience some "value added" and see a lot more unique content, you should also be on our web site. It turns out that in an average year we typically post about 1,500 pages of completely new and unpublished design projects and white papers on our web site. Go for it.

steve.ciarcia@circuitcellar.com

DigiView

100 MHz, 18 Channel, Portable Logic Analyzer

Auto Hardware Compression Captures
from 128K to 30 Billion Samples
@10ns resolution
only $499.00



Now with Serial Signal Analysis

Download and try the Software!

Cables & Micro-Clips Included

Professional Capture & Analysis Software Included

- Automatic Real-time Hardware Compression can save up to 5 minutes of data @ 10ns.
- Display I2C, Synchronous (SPI), Asynchronous (RS-232), State, Boolean, Bus and Analog Data.
- Edge and Pattern Triggers.
- Pattern Searches with Match, Duration & Animation.
- Specialized Sequential Searches for Serial and State Mode Signals.
- Drag & Snap Markers.
- Dual Waveform View with signal edge snapping.
- Multi-Signal Data Tables with Link Groups.
- Specialized Exports and List Views for Serial Signals.
- Print or Save Images with custom comments.
- USB Powered for Portability.
- USB 1.1 & 2.0 compatible.



PICmicro® MCU Programmer

Multi-Function In-Circuit & Gang Operation

only $199.00

- Supports 10, 12, 14, 16 & 18F Series PICmicro MCUs.
- Edit Code, EE Data, IDs and Configuration Fuses.
- Auto or Manual Serialization in Code or EE Data.
- Firmware auto synchronizes to Software version.
- Control Files protect Option Settings.
- Accepts Command-Line Parameters.
- Tests and Monitors Voltages.
- Control Signal for Self-Powered Circuits.
- Single and 4 Gang Adapters available.
- Includes Cables, Software & U.S. Power Supply.

EconoROM III™



EPROM & FLASH Emulation at its best!

from $179.00

FlexROM III™







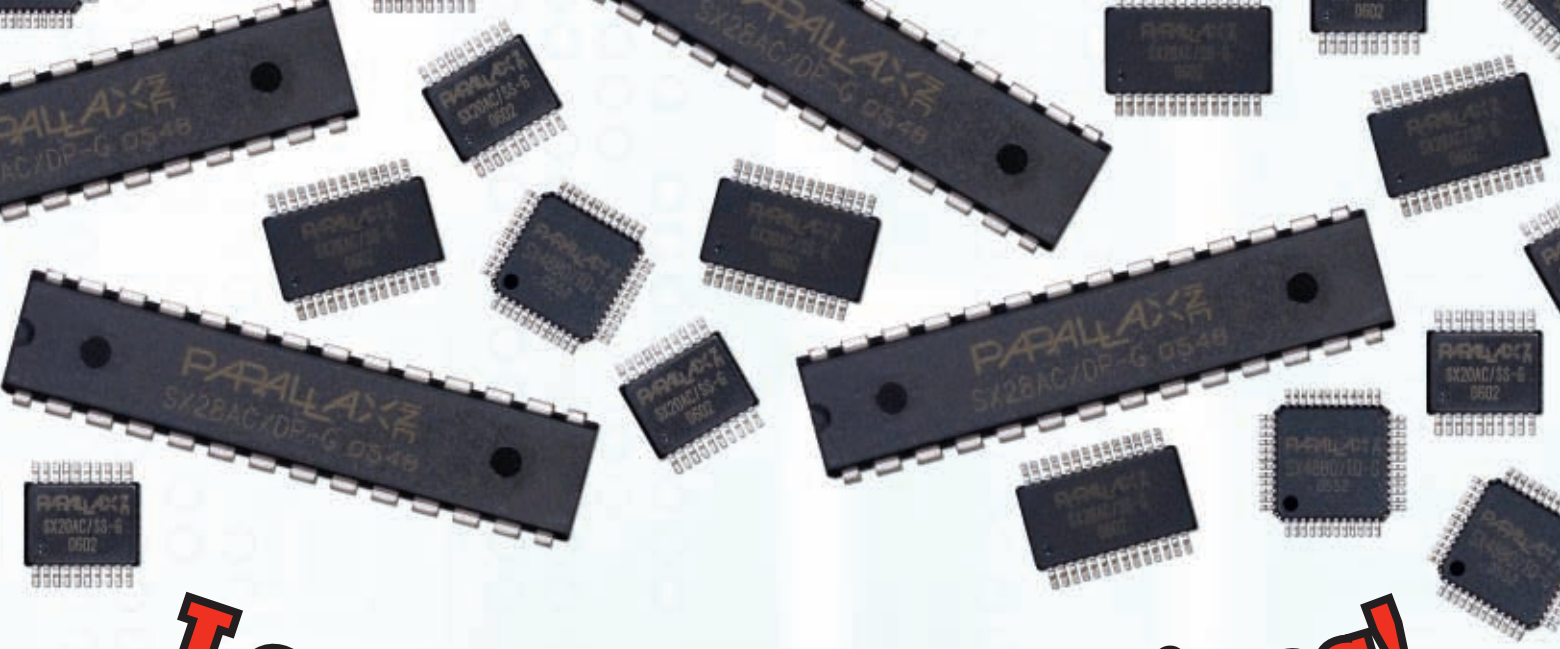
QuickLoader™

TechTools

www.tech-tools.com

(972) 272-9392 • sales@tech-tools.com

Copyright © 2006 TechTools • DigiView, FlexROM, EconoROM and QuickWriter are trademarks of TechTools • PICmicro is a registered trademark of Microchip Technology Inc.



Lower SX Chip Prices!

We have lowered our quantity pricing on SX chips. Now you can continue to use SX microcontrollers in your production projects and enjoy even better pricing. Parallax SX microcontrollers are RoHS Compliant, RISC architecture, high-speed microcontrollers with flash program memory, in-system programming and debugging capability.

Part #	Pins	I/O	EE/Flash	RAM	Qty. 1	Qty. 5	Qty. 100	Qty. 1000
SX20AC/SS-G	20	12	2K bytes	137 bytes	$2.79	$2.51	$2.23	$1.89
SX28AC/DP-G	28	20	2K bytes	136 bytes	$2.79	$2.51	$2.23	$1.89
SX28AC/SS-G	28	20	2K bytes	136 bytes	$2.79	$2.51	$2.23	$1.89
SX48BD-G	48	36	4k x 12 words	262 bytes	$2.79	$2.51	$2.23	$1.89

If you have not yet tried programming with an SX, this is the time to get started. Parallax offers free development software, including SX/B, a BASIC language compiler for the SX microcontroller. The SX/B compiler speeds the programming of SX chips by providing a simple, yet robust high-level language familiar to Parallax customers designed to help the transition from high-level programming (ie. BASIC Stamp®) to low-level programming (assembly language). *For beginners we recommend the SX Tech Tool Kit PLUS (#45181; $99.95) and a 7.5 VDC 1 Amp power supply (#750-00009; $10.95).*

Order Parallax **SX microcontrollers** online at www.parallax.com or call the Parallax Sales Department toll-free 888-512-1024 (Mon-Fri, 7am-5pm, PT).

PARALLAX

www.parallax.com

Parallax, and the Parallax logo are trademarks of Parallax, Inc. BASIC Stamp is a registered trademark of Parallax Inc.