

#208 November 2007

www.circuitcellar.com

CIRCUIT CELLAR®

THE MAGAZINE FOR COMPUTER APPLICATIONS

ANALOG TECHNIQUES

Switch Analog & VoIP Calls

DDS RF Signal
Generator

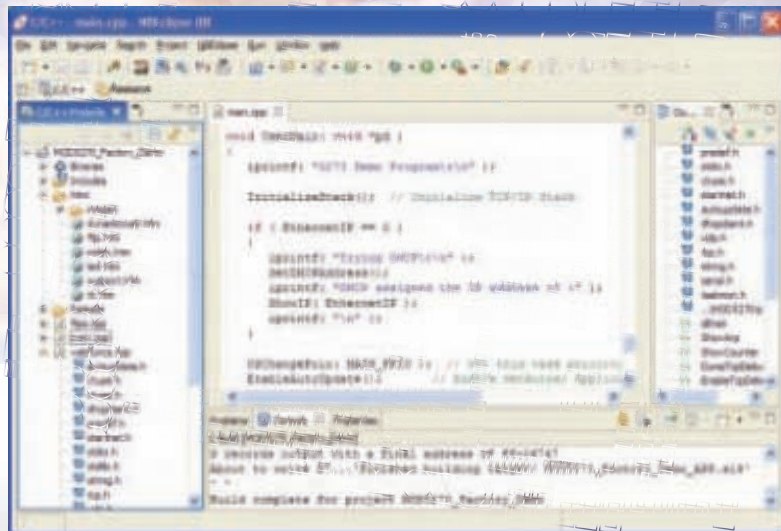
SMT Manufacturing
Basics

iEthernet Bootcamp



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

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



Product No. | NNDK-MOD5270LC-KIT
Information and Sales | sales@netburner.com
Telephone | 1-800-695-6828



It's showtime.

Get Your Ticket to the Ultimate Embedded Design Starter Kit.

Fixed-function microcontrollers had their share of the spotlight. It's time for a revolutionary—and simplified—approach to embedded application development. Get a Cypress PSoC® FirstTouch™ Starter Kit now and discover how much PSoC mixed-signal arrays—powerful, programmable analog and digital blocks, embedded memory and a fast MCU—shorten your time-to-market. This kit includes the easy-to-use PSoC Express™ visual embedded system design tool, and gives you embedded designs you can evaluate right out of the box. Get yours and step into the spotlight today.



Includes four ready-to-use mixed-signal applications on a single platform.

Buy your PSoC FirstTouch Ultimate Starter Kit now at:
www.cypress.com/go/FirstTouch

Buy Now
\$29.95*



Cypress, the Cypress logo and PSoC are registered trademarks, and Programmable System-on-Chip, PSoC Express, and FirstTouch are trademarks of Cypress Semiconductor Corporation. All other trademarks are properties of their respective owners. ©2007 Cypress Semiconductor Corporation. All rights reserved. *Does not include any applicable sales tax, shipping and handling costs. CPFAP001E



Link Instruments

PC-Based Test Equipment

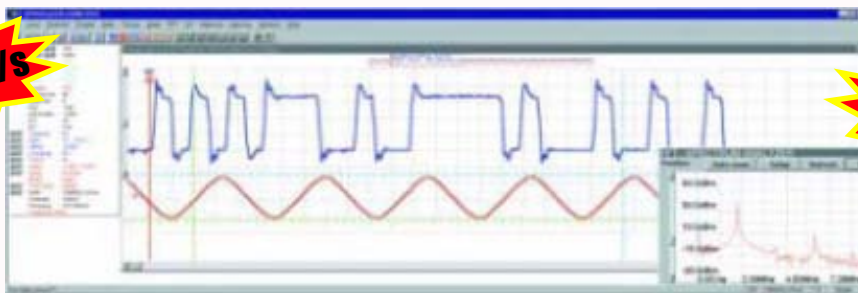
Digital Oscilloscopes



- 2 Channel Digital Oscilloscope
- **500 MSa/s** max single shot rate
- 1Mpts sample memory
 - 250 MSa/S (Dual channel) 512 Kpts
 - 500 MSa/S (Single channel) 1 Mpts
- Only 9 oz and 7" x 3.5" x 1.5"
- Portable and Battery powered
- USB 2.0
- FFT Spectrum Analyzer

DSO-8202 (200MSa,128K) **\$799**
 DSO-8502 (500MSa,1Mpt) **\$950**

500MSa/s

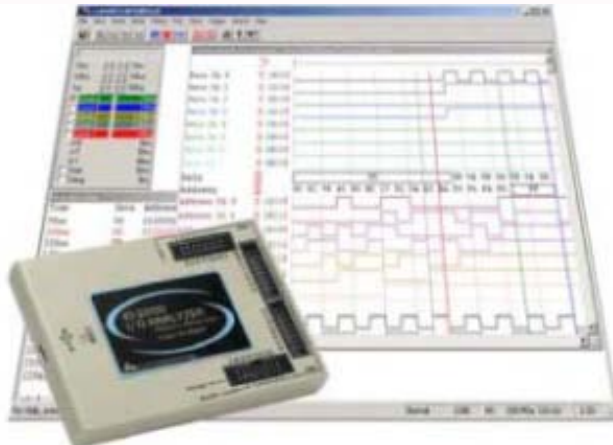


1Mpts

Logic Analyzer & Pattern Generator

NEW!

Portable
3.2" x 3" x 0.65"
USB 2.0 Powered



- Logic Analyzer (32 channels)
- Pattern Generator (up to 32 channels)
- up to 400 MSa/s
- Variable Threshold
- 2 External Clocks
- SPI output and disassembly
- I²C output and disassembly
- up to 2Msamples/ch

IO-3208A **\$750**
 IO-3232A **\$899**
 IO-3232B **\$1399**



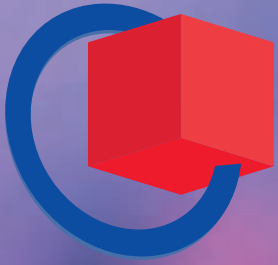
Link Instruments (973) 808-8990

www.Linkins4.com

Note the date!

Nuremberg, Germany

February 26 – 28, 2008



embedded world 2008

Exhibition & Conference Nürnberg

...it's a smarter world

www.embedded-world.de

Exhibition organizer
NürnbergMesse
Tel +49 (0) 9 11.86 06-0
info@nuernbergmesse.de

Conference organizer
DESIGN&ELEKTRONIK
Tel +49 (0) 81 21.95-13 40
cgrote@design-elektronik.de

Media partners

Markt & Technik
Die unabhängige Wochenzeitung für Elektronik

Computer & AUTOMATION
Fachmagazin der Fertigungs- und Prozesstechnik

elektronik report

DESIGN & ELEKTRONIK
• PRINT • ONLINE • KONGRESSE •

Elektronik automotive
Fachmagazin für Entwicklungen in der Kfz-Elektronik und Telematik

elektronik net.de

Elektronik
Fachzeitschrift für Industrie- & Amateurfunk Elektronik

Elektronik wireless
Fachmagazin für Entwicklungen in drahtlosen Systemen

NÜRNBERG MESSE

TASK MANAGER

New Initiatives

I enjoy attending conferences because I get to leave the editorial office and meet face to face with our readers, writers, advertisers, and contest sponsors. In September, I drove to the Embedded Systems Conference in Boston. As usual, I had a great time meeting with everyone who stopped by our booth.

This year's conference was important for a few reasons. Most notably, we launched two new exciting initiatives: the WIZnet iEthernet Design Contest 2007 and a call for articles about "intelligent energy" (IE) solutions.

As for the WIZnet contest, check out Fred Eady's contest primer on page 34. Fred introduces you to the power of the WIZnet W5100 hard-wired TCP/IP Ethernet controller. If you're inspired, you can easily join the Ethernet revolution at www.circuitcellar.com/wiznet/. Registering for a project number will take only a few minutes. With \$15,000 in total cash prizes up for grabs, it's well worth the effort.

Next month, we will run a new section of the magazine called Intelligent Energy (IE) Solutions. To kick things off, the new section will feature Part 1 of Steve Ciarcia's series about his new photovoltaic (PV) installation. As you know, Steve has mentioned his PV system in his monthly Priority Interrupt columns. Next month, you'll get to read about the details.

The purpose of the IE section is to highlight exciting new designs that address topics such as solar-powered systems, energy-efficient embedded designs, and power-saving applications. As many of you learned in Boston, we're challenging you to draw up an IE design, build a system, and send us an article proposal. If your proposal intrigues us, we might ask you to submit an article for this new section. Are you up for the challenge?

As you prepare to enter the WIZnet contest or start an IE design (or both), be sure to read through this issue's batch of compelling articles. Our authors cover a wide variety of well-designed applications.

On page 12, David Rowe describes how he combined his interest in embedded systems and telephony to design a μ Clinux-powered IP-PBX. If you follow his lead, you too will be able to switch analog and VoIP calls.

Want to know what it took to win a top prize in the Luminary Micro DesignStellaris2006 Contest? Check out Thomas Alldread's versatile NimbleSig design on page 22.

In his well-known ECE 576 course at Cornell University, Bruce Land and his students use FPGAs to build embedded systems. In "Hybrid Computing on an FPGA" (p. 44), Bruce explains how he simulates the parallel functions of an analog computer on an FPGA.

Because we like variety, the second half of this issue features articles on topics that are a bit less hardware-centric: communication protocols (p. 50), software design techniques for small embedded systems (p. 58), and product development (p. 63). We're confident that the advice provided in these three articles will serve you well as you start your next project. Good luck!

cj@circuitcellar.com



CIRCUIT CELLAR®

THE MAGAZINE FOR COMPUTER APPLICATIONS

FOUNDER/EDITORIAL DIRECTOR

Steve Ciarcia

CHIEF FINANCIAL OFFICER

Jeannette Ciarcia

MANAGING EDITOR

C.J. Abate

MEDIA CONSULTANT

Dan Rodrigues

WEST COAST EDITOR

Tom Cantrell

CUSTOMER SERVICE

Debbie Lavoie

CONTRIBUTING EDITORS

Jeff Bachiochi

Ingo Cyliax

Robert Lacoste

George Martin

Ed Nisley

CONTROLLER

Jeff Yanco

ART DIRECTOR

KC Prescott

GRAPHIC DESIGNER

Mary (Turek) Sobuta

STAFF ENGINEER

John Gorsky

PROJECT EDITORS

Steve Bedford

Ken Davidson

David Tweed

ASSOCIATE EDITOR

Jesse Smolin

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.

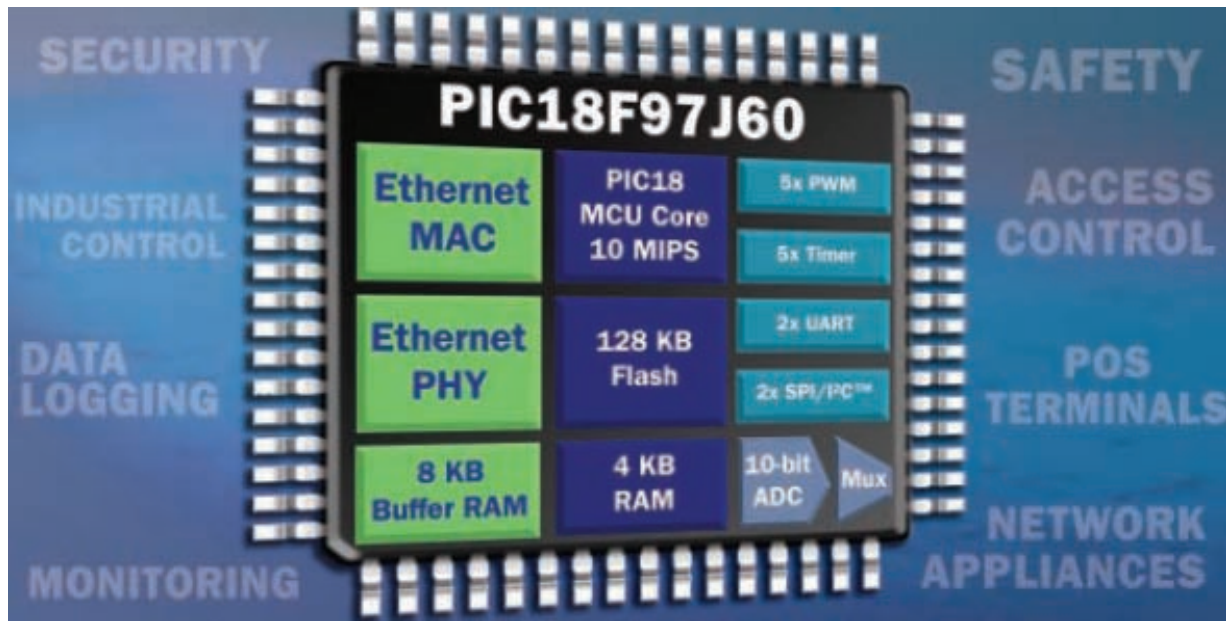
Circuit Cellar® makes no warranties and assumes no responsibility or liability of any kind for errors in these programs or schematics or for the consequences of any such errors. Furthermore, because of possible variation in the quality and condition of materials and workmanship of reader-assembled projects, 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.

Embedded Ethernet

Any Application, Any Location



Monitor, control or re-program your application remotely with ease using the integrated Ethernet PIC18F97J60 family and **FREE** TCP/IP software.

GET STARTED IN 3 EASY STEPS at www.microchip.com/ethernet.

1. Learn about our Ethernet devices in 20 minutes
Take advantage of our Ethernet web seminars.

2. Download our FREE TCP/IP software
Our TCP/IP stack is available in source code for flexible and optimized code size.

3. Check out our low-cost Ethernet tools
Evaluate the PIC18F97J60 family with the **PICDEM.net™ 2 Demo Board** – available for only \$129.98 when you use coupon code **ETHTOOL** during checkout at www.microchipDIRECT.com.



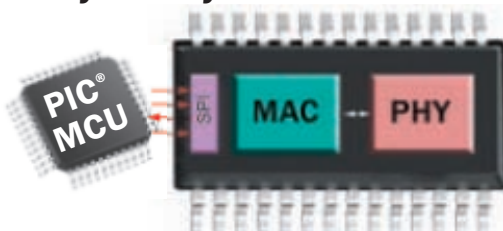
Purchase and program your Ethernet PIC® microcontrollers and related development tools at...

microchip
DIRECT
www.microchipsdirect.com

Device	Pins	Flash (KB)	Features
PIC18F97J60	100	128	10-BaseT Ethernet 12 KB RAM (8 KB dedicated Ethernet)
PIC18F87J60	80	128	
PIC18F67J60	64	128	
PIC18F96J65	100	96	5x 16-bit timers
PIC18F86J65	80	96	10-bit ADC, 16 channels
PIC18F66J65	64	96	2 analog comparators
PIC18F96J60	100	64	2 UART with LIN protocol
PIC18F86J60	80	64	2 SPI, 2 I ² C™
PIC18F66J60	64	64	Industrial Temperature -40° to +85°C
ENC28J60	28	8K RAM	MAC, PHY, SPI Interface

Visit www.microchip.com/ethernet for additional devices supporting connectivity solutions and related development tools, application notes, reference designs, web seminars and more!

Or you may consider...



Adding Ethernet to any application with Microchip's **ENC28J60** stand-alone Ethernet controller with full software support for PIC18, PIC24 and dsPIC® DSCs.



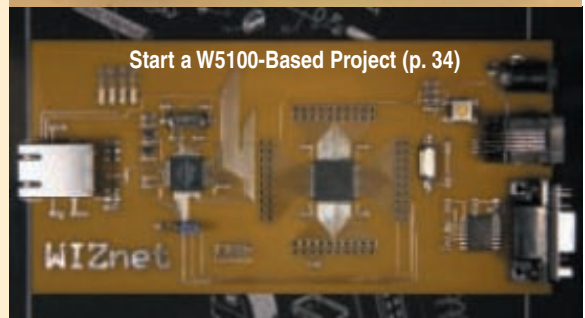
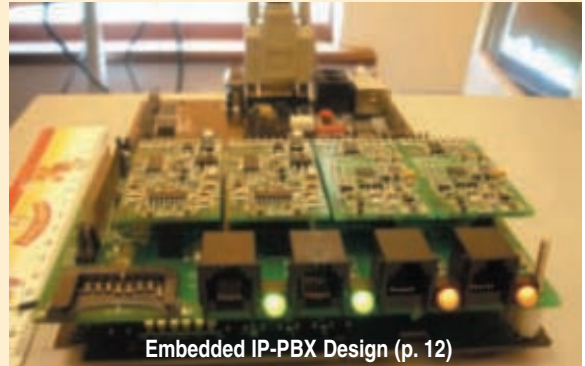
MICROCHIP

www.microchip.com/Ethernet

November 2007: Analog Techniques

FEATURES

- 12 **Embedded IP-PBX**
Switch Analog and VoIP Calls
David Rowe
- 22 **NimbleSig**
A Compact DDS RF Signal Generator
Thomas Alldread
Second Prize Luminary Micro DesignStellaris2006 Contest
- 34 **iEthernet Bootcamp**
Get Started with the W5100
Fred Eady
WIZnet iEthernet 2007 Design Contest Primer
- 44 **Hybrid Computing on an FPGA**
Bruce Land
- 50 **Communication Protocols**
Massimo Manca
- 58 **Resilience in Embedded Designs (Part 3)**
Software
Aubrey Kagan
- 63 **SMT Manufacturing**
Take a Board from Prototype to Production
Zack Gainsforth



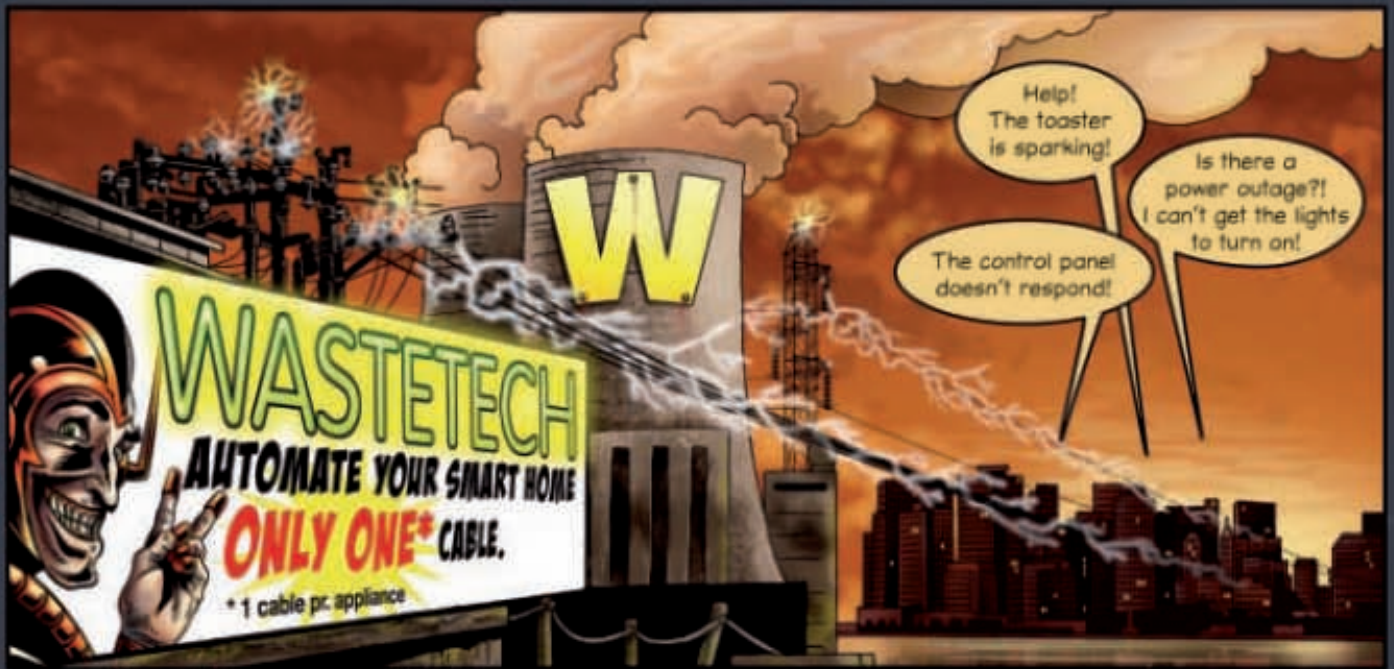
COLUMNS

- 70 **FROM THE BENCH**
Motor Driving for a Robotic Arm
Jeff Bachiochi
- 76 **LESSONS FROM THE TRENCHES**
A Flash in the Pan
George Martin
- 80 **SILICON UPDATE**
Thanks for the MEMS
Tom Cantrell



DEPARTMENTS

- 4 **TASK MANAGER**
New Initiatives
C.J. Abate
- 8 **NEW PRODUCT NEWS**
edited by *John Gorsky*
- 93 **CROSSWORD**
- 94 **INDEX OF ADVERTISERS**
December Preview
- 96 **PRIORITY INTERRUPT**
Let There Be Light
Steve Ciarcia



Go wireless with AVR Z-link

Get more at: www.atmel.com/AVRman

© 2007 Atmel Corporation. All rights reserved. Atmel, AVR and logo 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 Mylo and Farnes Fabrics. All 2007.



2.7-W CONSTANT OUTPUT POWER CLASS-D AMPLIFIER

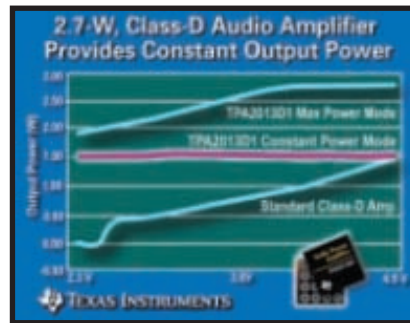
The TPA2013D1 is a monolithic, filter-free, class-D audio power amplifier with an integrated boost converter that provides constant output power for portable applications such as personal navigation devices, PDAs, mobile phones, portable media players, and handheld gaming devices. The amplifier generates high output power from a low supply voltage without distorting the audio and can also supply the power of external devices. The device has a wide supply voltage operation of 1.8 to 5.5 V to simplify power supply design and allow for direct connection to the battery. The new device provides a maximum output power of 2.7 W across a 4-Ω load or 2.2 W across an 8-Ω load in addition to providing an adjustable constant output power of up to 1.5 W in the entire Lithium-Ion battery range of 2.3 to 4.8 V. This capability makes the audio output power insensitive to bat-

tery voltage fluctuations, thereby maintaining audio quality and volume as the battery discharges.


Several key features of the TPA2013D1 help increase system audio quality. For example, all internal modules run off the same reference clock, silencing potential audible beat frequencies that could occur when using a separate discrete amplifier and boost converter. The synchronized clock and very high power supply rejection ratio, 95 dB at 217 Hz, serve to further reduce noise in the system.

The WCSP costs \$1.55 in 1,000-unit quantities and \$1.45 for the QFN package.

Texas Instruments, Inc.
www.ti.com




STATEMENT REQUIRED BY THE ACT OF AUGUST 12, 1970, TITLE 39, UNITED STATES CODE SHOWING THE OWNERSHIP, MANAGEMENT AND CIRCULATION OF CIRCUIT CELLAR, THE MAGAZINE FOR COMPUTER APPLICATIONS, published monthly at 4 Park Street, Vernon, CT 06066. Annual subscription price is \$23.95. The names and addresses of the Publisher, Editorial Director, and Managing Editor are: Publisher, Sean Donnelly, 4 Park Street, Vernon, CT 06066; Editorial Director, Steven Ciarcia, 4 Park Street, Vernon, CT 06066; Managing Editor, Carmine J. Abate, 4 Park Street, Vernon, CT 06066. The owner is Circuit Cellar, Inc., Vernon, CT 06066. The names and addresses of stockholders holding one percent or more of the total amount of stock are: Steven Ciarcia, 4 Park Street, Vernon, CT 06066. The average number of copies of each issue during the preceding twelve months is: A) Total number of copies (net press run) 22,801; B) Paid/Requested circulation (1) Mail subscriptions: 9,327; (2) Sales through dealers and carriers, street vendors and counter sales: 5,249; C) Total paid/requested circulation: 14,576; D) Nonrequested distribution (by mail and outside the mail) (1) Outside county nonrequested (sample copies): 2,577; (2) Nonrequested copies distributed outside the mail (trade shows, other sources) 1,922; E) Total nonrequested distribution: 4,499; F) Total distribution: 19,075; G) Copies not distributed: 3,726; H) Total: 22,801; I) Percent paid/requested: 76.4%. Actual number of copies of the single issue published nearest to filing date (October 2007, Issue #207): A) Total number of copies (net press run) 26,630; B) Paid/Requested circulation (1) Mail subscriptions: 9,658; (2) Sales through dealers and carriers, street vendors and counter sales: 5,173; C) Total paid/requested circulation: 14,831; D) Nonrequested distribution (by mail and outside the mail) (1) Outside county nonrequested (sample copies): 1,793; (2) Nonrequested copies distributed outside the mail (trade shows, other sources) 6,355; E) Total nonrequested distribution: 8,148; F) Total distribution: 22,979; G) Copies not distributed: 3,651; H) Total: 26,630; I) Percent paid/requested: 64.5%. I certify that the statements made by me above are correct and complete. Sean Donnelly, Publisher.




HOME MONITORING AND CONTROL OVER SKYPE™

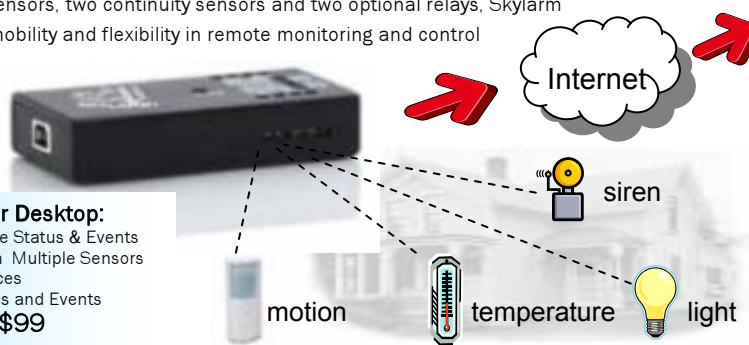
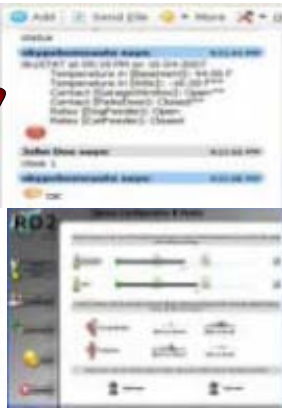
Skype - based Alarm Messenger



SKYLARM™ is a small USB device and GUI software that allows a user to perform remote premises monitoring and appliance control via Skype Communicator. Equipped with two temperature sensors, two continuity sensors and two optional relays, Skylarm provides unparalleled mobility and flexibility in remote monitoring and control



Designed and Made in USA

From any Skype Mobile or Desktop:

- Get Temperature Status & Events
- Get Alarms from Multiple Sensors
- Control Appliances
- Configure Alarms and Events
- Prices start at **\$99**

FEATURES & SPECIFICATIONS		RD2TECH	
Temperature Sensors: 2	Alarm Types: 5	Supported OS: Windows 2000, XP Vista	183 Matlook Place Somerset, NJ 08873, USA Phone: +1 732 481 - 5243 Fax: +1 732 873 - 5465 Email: info@rd2tech.com Web: www.rd2tech.com
Temperature Range: -10..+140F	Auxiliary Sensor Types: Humidity Water Pressure	Number of Users: Unlimited	
Dry Contact Sensors: 2		Software required: Skype	
Relays' Outputs (option): 2			
Computer I/O required: USB			

Visit WWW.RD2TECH.COM to ORDER!

NEW PRODUCT NEWS

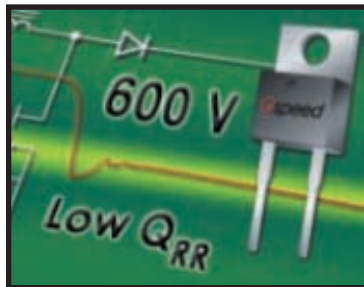
NEW RECTIFIERS BOOST PFC EFFICIENCY

The new **Q-Series** rectifier family of devices is designed specifically for use in power factor correction (PFC) circuits that deliver 200 to 1,000 W of power. Replacing an ultrafast diode with a Q-Series device typically realizes efficiency improvements of 0.5 to 1.5% in the boost converter stage, which are similar to those obtained by using exotic silicon carbide diodes. Developed using a proprietary technology, the Q-Series family of devices has the lowest reverse-recovery-charge (Q_{RR}) and the best softness factor of any high-speed silicon rectifiers currently available on the market.

The Q-Series product family consists of three devices, which are rated to deliver continuous forward current values of 3, 5, and 8 A, respectively. The 3-A device (LQA03TC600) would typically be used in boost converter stages that deliver between 200 and 400 W of continuous output power. The 5-A device (LQA05TC600) would find its home in converters that deliver between 400 and 700 W, while the zone for the 8-A device (LQA08TC600) is from about 700 to 1,000 W.

The LQA03TC600 costs \$0.78 each, the LQA05TC600 costs \$0.88 each, and the LQA08TC600 costs \$1.08 each in 50,000-piece quantities.

Qspeed Semiconductor
www.qspeed.com



EASY-TO-PROGRAM LINUX CONTROLLER

The **Omniflash** controller provides users with a rich array of I/O devices seamlessly supported by a preinstalled Linux 2.4 kernel. The controller comes furnished with 10/100 Ethernet, two serial ports, a battery-backed clock/calendar, a USB, digital I/Os, and stereo audio line level outputs. The 200-MHz ARM9 processor handles complex multitasking operations efficiently. Onboard memory includes 16 MB of flash memory organized as an Ext2 file system and 32 MB of SDRAM.

The Linux operating system also includes more than 150 standard Linux/Unix system utilities including ftp, tftp, telnet, and vi. Also included in the development kit is a bootable Knoppix CD-ROM preconfigured with development tools to support the Omniflash.

The Omniflash controller costs \$129 each in quantities of 100. Development kits are available for \$199.



JK microsystems, Inc.
www.jkmicro.com

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

Order Online At
rabbitappkits.com

WIRELESS MADE SIMPLE

BRING YOUR PRODUCT QUICKLY AND LEGALLY TO MARKET

RF Modules

Low-Cost TX & RX Modules

Multi-Channel Modules

Long-Range Modules

Add INSTANT wireless analog / digital capability to your product.

OEM Products

Handheld TXs

Keyfob TXs

FCC PRE-CERTIFIED & ready to customize for your application

Function Modules

Antennas

From ceramic chips to gain Yagi, keyless entry to Wi-Fi.

Specialty

GPS

Embedded Chips

Permanent Mount

Whips

Magnetic Base

Gain Antennas

Gain Antennas

800-736-6677
159 Ort Lane · Merlin, OR 97532
www.linstechnologies.com

www.circuitcellar.com

CIRCUIT CELLAR®

Issue 208 November 2007 9

NEW PRODUCT NEWS

METRIC THREADED EMI FILTERS AVAILABLE IN FEED-THROUGH AND PI CIRCUITS

New metric threaded EMI filters from Spectrum Control include an M3 X 0.5 feed-through circuit and an M4 X 0.7 Pi circuit. These filters are easily mounted in a tapped hole or through-hole and have a hex head dimension across the flats of 4 mm. The metric threading makes these EMI filters ideal for design solutions in European and Asian markets.

Spectrum's metric threaded EMI filters have capacitance values ranging from 100 to 12,000 pF, a current rating of 10 A, a voltage rating of 100 VDC, and a dielectric withstanding voltage of 250 VDC. The durable resin-sealed case provides excellent environmental protection. They are primarily used in filtering signal/data lines, industrial controls, telecommunications, medical equipment, and test equipment.

The metric threaded EMI filters cost from \$2 to \$5 depending on configuration and quantity, with a lead time of eight weeks.



Spectrum Control, Inc.
www.specemc.com

NEW C COMPILER FOR PIC24 MCUs AND dsPIC DSCs

The PCD is a low-cost, quality 24-bit C compiler for the PIC24 MCUs and dsPIC DSCs. The compiler includes peripheral libraries for SPI, I²C, UART, RTC, input capture, and PWMs. The PCD includes new built-in functions and examples to aid in getting a project started.

CCS will also offer prototyping boards to aid in PIC24 and dsPIC DSC development. The DSP starter development kit features a dsPIC30F2010 prototyping board. The DSP starter development kit is a general-purpose development board including a potentiometer, a push button, three LEDs, an RS-232 level converter, and an ICD connector and header to access the dsPIC30F2010 chip. CCS will also offer two PIC24 development kits and DSP analog/audio development kits by the end of the year. The DSP analog/audio board will demonstrate the DSP features of the dsPIC by providing an example audio conditioning board. Kits may be purchased with a PCD compiler or as a board-only option.

The PCD is in the fundamental development period and is now available for sale. The PCD is offered as a command-line compiler for \$250, an add-on to an existing PCWH compiler for \$150, a stand-alone IDE compiler for \$350, and a full-version PCWHD compiler for \$575.

Custom Computer Services, Inc.
www.ccsinfo.com



cleverscope plug & play instruments

why is everyone talking about cleverscope?*



cleverscope CS328A

visit our website and discover for yourself

or contact our USA master distributor



www.cleverscope.com

* quote: "fantastic instrument—solved very difficult waveform observations"

EzPCB
Professional PCB Supplier

High Quality
Competitive Price

URL: www.EzPCB.com
Email: sales@ezpcb.com Tel: +86 139 1002 1704
HDI Up To 50 Layers, 2.5mil T/C, 0.1mm Hole Size
Other Products: Stencils, Keypads, Frontpanels, Flex PCB, Enclosures, Turnkey Services

NEW PRODUCT NEWS

FAMILY OF HANDHELD DIGITAL MULTIMETERS

The U1240A series of digital multimeters enables field or facilities technicians and engineers to do more than just voltage, current, and resistance measurements. For instance, providing more than typical voltage measurements on switches, the U1240A series enables users to observe open/close switch behaviors in the presence of intermittent signals.

Users can also easily gauge the extent of transformer heating or the efficiency of refrigeration systems with the DMM's dual and differential temperature capabilities. This, coupled with its microamp measurement function, is especially handy for troubleshooting HVAC equipment and sensors.

When maintaining AC motors or troubleshooting circuit breakers that trip prematurely, the series helps users quickly verify the presence of harmonics that may have caused device overheating. With its internal memory, the DMM lets users conveniently perform data collection on the go for later analysis.

The U1240A series of handheld digital multimeters comprises two models: the U1241A and the U1242A. They are durable, safe, and easy to use. These DMMs simplify installation and maintenance by providing a broad range of measurement functions, including voltage, capacitance, and temperature. Key features include 10,000-count resolution, dual display, dual-intensity backlight, dual and differential temperature, harmonic ratio, a switch counter, and data logging.

The Agilent U1240A series also provides wider usage for higher resistance measurements (up to 100 M Ω) and lower current measurements (down to 0.1 mA).

The Agilent U1240A series starts at \$200.

Agilent Technologies, Inc.
www.agilent.com



10-MHz DIGITALLY PROGRAMMABLE INSTRUMENTATION AMPLIFIER

Designed for data acquisition systems, automatic test equipment, and biomedical instruments requiring fast, accurate measurement with robust signal conditioning over large voltage ranges, the AD8253 digitally programmable instrumentation amplifier is able to reach gains of 1,000 while offering DC precision and AC bandwidth unmatched by other instrumentation amps or discrete solutions on the market. A large gain setting allows small signals, such as those from sensors, to be amplified to drive an ADC.

The AD8253 is digitally programmable with gain settings of 1, 10, 100, and 1,000, letting users adjust gain even after the devices are designed into the system. Operating from ± 5 to ± 15 V, it achieves 10-MHz bandwidth and 0.5- μ s settling time to 0.01%. The input offset drift of 1.2 μ V/ $^{\circ}$ C and gain drift of 10 ppm/ $^{\circ}$ C is the lowest of any programmable gain in-amp. Manufactured using the iCMOS process technology, the AD8253 draws just 4 mA of quiescent current and is available in 10-lead MSOP (mini small-outline plastic) packages.

The AD8253 costs \$4.95 per unit in 1,000-unit quantities.

Analog Devices, Inc.
www.analog.com



Eval Versions Available



- Customer Sample (Aviation Device)

GUI (emWin[®])

+++ 8/16/32 bits +++

- 2D Graphic Library
- Window Manager/Widgets
- Complete ANSI "C" Source Code
- RTOS Independent
- Font Converter
- Image Converter



- Customer Sample (Automotive Device)



Develop your fully functional application on a PC. Simulating everything from hardbuttons to a touchscreen interface.

- emWin PC Simulation Environment

- Detailed User's Manual
- Extensive Sample Code
- PC Simulation Included

emWin is designed to provide an efficient, processor independent, LCD controller independent, graphical user interface for use in any embedded application, with or without a commercial RTOS.



- Customer Sample (Medical Device)

www.emWin.info

www.segger.com

Embedded IP-PBX

Switch Analog and VoIP Calls

David Rowe describes the design of a μ Clinix-powered IP-PBX capable of switching both analog and VoIP calls. With an Analog Devices Blackfin processor, some custom hardware, and Asterisk PBX software, you can build a similar system.

I have always been fascinated with embedded systems and telephony. I really like working “close to the machine” on embedded systems, and I have a parallel interest in speech and telephony that dates back to my ham radio days as a teenager. One project I had been dreaming about for years is an embedded telephony platform for VoIP or IP-PBX applications.

One problem with telephony is that it requires a lot of processing power. Unfortunately, most embedded processors are not too powerful. So, a common design approach for embedded telephony consists of a general-purpose microcontroller combined with a special-purpose DSP chip to handle the hard-core number crunching. This two-CPU approach increases the system cost and complexity.

About 18 months ago, I stumbled across the Blackfin processor from Analog Devices. The Blackfin is a powerful host processor and a DSP (i.e., it can run μ Clinix applications and DSP code efficiently and simultaneously on one processor).

A typical embedded processor will struggle to run more than one or two channels of G.729 speech compression or echo cancellation software. The Blackfin, running at 500 MHz, is capable of running 60 channels of G.729 speech compression or echo cancellation. This makes it possible to build low-cost, high-performance embedded telephony systems, like an IP-PBX that can support multiple analog and VoIP channels at the same time. For about \$500, you can

now build an IP-PBX with features that only \$10,000 PBXs had a few years ago.

This work is part of the free telephony project (www.rowetel.com/ucasterisk). The goal of the project is to build open-source telephony hardware and software (e.g., an embedded Asterisk IP-PBX) at a low cost. Both the hardware and software are open source. You are free to copy and reuse the hardware designs.

IP-PBX OVERVIEW

What is an IP-PBX and how does it work? Figure 1 is a block diagram of the embedded IP-PBX that shows the major hardware and software components.

There are two types of analog ports: FXO ports that connect to your local telephone exchange and FXS ports that connect to analog handsets. This jargon can be confusing. The way I remember is the “S” in FXS stands for “station” handset.

The analog ports convert the voice and signaling information to digital signals that the IP-PBX can process.

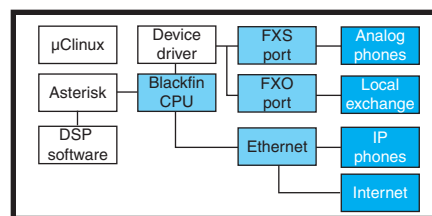


Figure 1—This is a block diagram of the Embedded IP-PBX showing the major software and hardware components. The heart of the system is the Blackfin processor. This is connected to hardware such as the FXO/FXS telephone ports and Ethernet. The Blackfin processor runs the μ Clinix operating system and Asterisk application software.

Examples of signaling information are the ring detection and on/off hook status signals. Analog ports are surprisingly difficult to build because they use a mixture of rather old technologies. For example, a ring signal may be 200 V_{pp} (at low current), on and off hook status is indicated by loop current flowing, and the transmit and receive audio is mixed together on just two wires, which leads to echo problems. All of this must be reliably and safely connected to low-voltage digital systems. Fortunately, there are excellent chipsets available to help build cost-effective and reliable analog interfaces.

Telephone calls can also flow through the Ethernet port using VoIP. Both local (e.g., using SIP phones) and trunked calls can be performed using VoIP. The magic of an IP-PBX is that analog and VoIP calls can be tied together. You can route a call from an analog handset over the Internet to save money on long-distance calls.

What happens when you make a regular analog phone call? First, pick up an analog phone’s handset. This generates an “off-hook” event in the FXS port that tells the Asterisk software to generate a dial tone in your phone. When you dial 9, DSP software decodes the DTMF tones and presents Asterisk with the digit. Asterisk then connects your FXS port to an FXO port so you can reach the local exchange where you can dial a phone number as usual.

What about VoIP calls? Well, say you use the same analog phone to make the call, but this time you dial

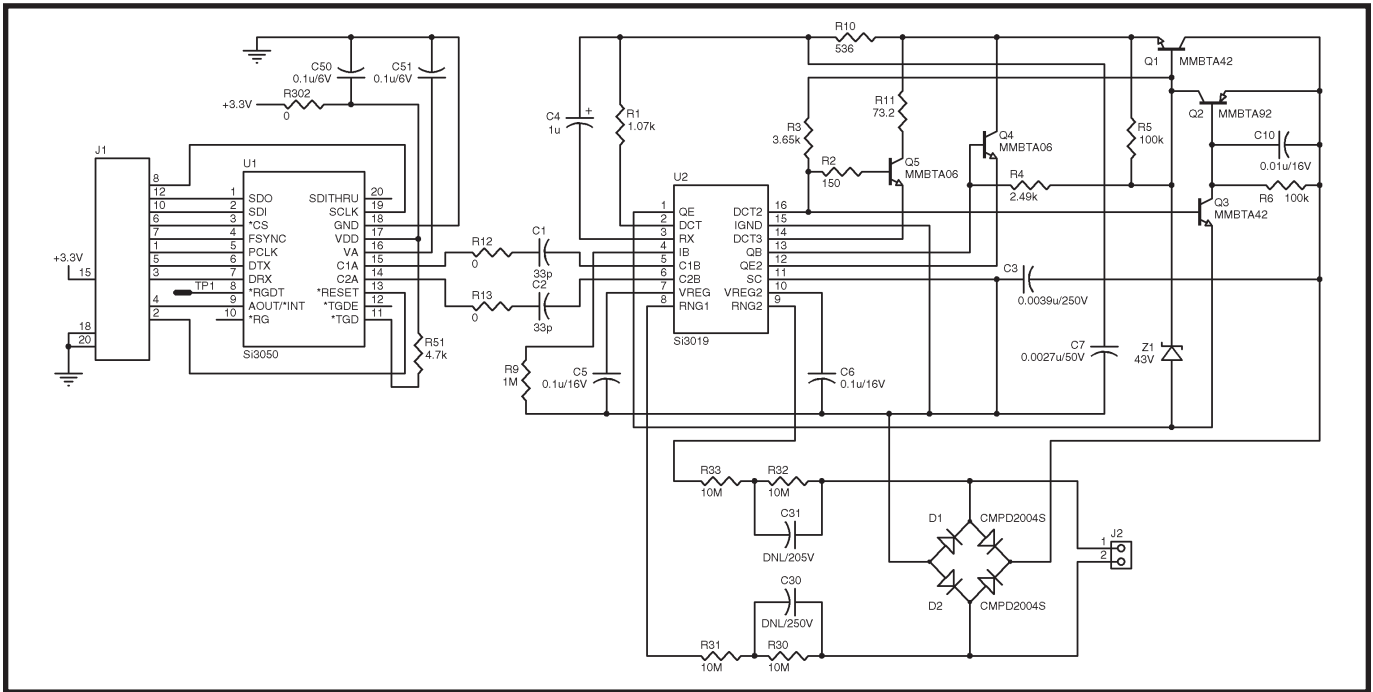


Figure 2—Take a look at the FXO module schematic. It provides an interface between a telephone line and the Blackfin processor. Its most important task is to provide high-voltage isolation, which is achieved using the capacitive barrier formed by C1 and C2.

long distance. In this case, Asterisk is programmed to route the call over the Internet. To save bandwidth, the DSP software compresses the speech samples from the FXS port from 64 kbps down to 8 kbps. The Asterisk software then puts the speech samples in packets and squirts them out the Ethernet interface onto the Internet.

A lot of the IP-PBX's power is provided by the Asterisk software, whose primary sponsor is Digium. The ability to run a powerful operating system such as μ CLinux is also very useful. For example, you can telnet into the IP-PBX



Photo 1—Here you see the 4fx daughter board and modules before assembly. Each of the four modules at the bottom mate with matching connectors in the middle of the 4fx daughter board. Note the large vacant area in the middle of the PCB. It is necessary to provide physical isolation between the FXO ports and the rest of the system.

while it is running to debug and configure it.

THE HARDWARE

The hardware is built around four PCBs. The first board is an Analog Devices Blackfin STAMP card. It is a development system that is available off-the-shelf from Digi-Key. It contains an Analog Devices ADSP-BF537 Blackfin chip, 64 MB of RAM, 4 MB of flash memory, and connectors that break out many of the ports. It runs μ CLinux and is supported by www.blackfin.uclinux.org.

One great feature is that an open-hardware/software community exists around the Blackfin. Reference designs for Blackfin hardware (such as the STAMP boards) and various daughter boards are freely available.

On top of the STAMP sits a 4fx daughter board. (Why are boards always girls?) It holds some glue logic and sockets for the modules and also supports an MMC to provide extra flash memory storage. The 4fx's name comes from the fact that the daughter board can support up to four FXS or FXO modules.

The modules are the little boards that plug into the daughter board. There are two types of modules, FXS and FXO. Photo 1 shows the 4fx

daughter board and modules disassembled. There are two modules of each type in the photo. Photo 2a features

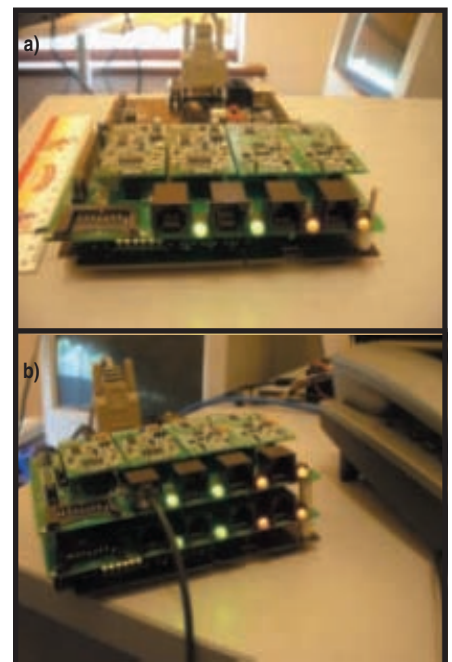


Photo 2a—The IP-PBX is assembled and configured for four analog ports. From top to bottom, you can see the FXS/FXO modules, the 4fx daughter board, and the BF537 STAMP. There is an optional MMC connector on the left. Green indicates an FXS port, red an FXO. **b**—The design is expandable to multiples of four ports by stacking additional daughter cards and modules. The wcfxs device driver automatically detects the number and type of ports available. In this case, there are four FXS and four FXO ports.

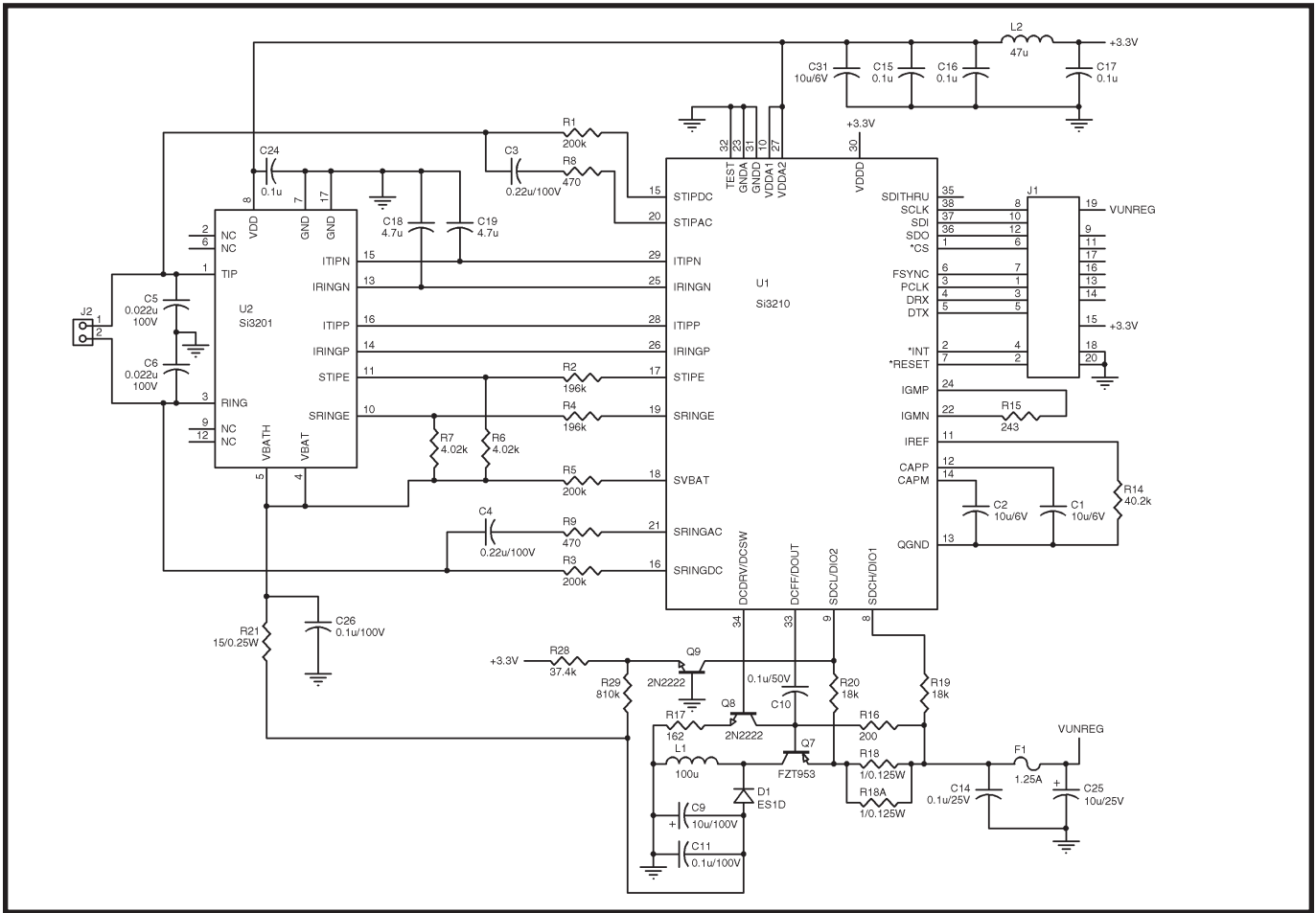


Figure 3—Take a look at the FXS module schematic. It provides an interface between a telephone and the Blackfin processor. The DC/DC converter in the lower-right corner generates 48-V battery and 200-V_{pp} ring voltages from a standard 12-V supply.

the entire system assembled, powered up, and ready to make calls. There are two FXS modules on the left and two FXO modules on the right. The color of each LED indicates the type of module inserted via an autodetection algorithm. It is also possible to stack multiple 4fx boards to expand the number of channels that can run on the system (see Photo 2b).

In the next few sections, I'll describe the schematic-level design for the FXS, FXO modules, and the 4fx card. The Blackfin STAMP card schematics are available at www.blackfin.uclinux.org.

FXO AND FXS MODULE DESIGN

The gEDA suite of GPL electronic design automation tools was used for schematic entry, PCB layout, and Verilog simulation throughout the project (www.geda.seul.org). The use of open-source tools makes it easy for anyone to modify the designs. Because the hardware designs are "open," it is nice to use open-source tools where possible.

The schematics for the FXS and FXO modules are shown in Figures 2 and 3. Both modules are based on chipsets from Silicon Laboratories. The circuit designs for the modules are derived from the reference circuits provided in the Silicon Laboratories datasheets.

The most important function of the FXO module is to isolate the "line side" of the port from the low-voltage digital side. There are safety reasons for the isolation. If lightning hits the phone lines, you will want a degree of physical isolation between the phone line and the rest of the hardware. This isolation is rigorously tested during FCC-68 compliance testing by the application of high voltages that simulate lightning. The Silicon Laboratories chips achieve it with a capacitive isolation barrier (C1 and C2). The Silicon Laboratories Si3050/Si3019 chipset used for the FXO module also performs other functions, such as line voltage and current monitoring, impedance matching, A/D

and D/A conversion of the speech signal, ring detection, and DC termination.

A screenshot of the PCB layout for the FXO module is posted on the *Circuit Cellar* FTP site. The PCB measures about 50 × 25 mm. Note the "isolation barrier" between the two chips, which only C1 and C2 are allowed to bridge. The PCB layout was carefully designed to keep the line side and low-voltage side physically isolated by a 3- to 4-mm gap. Even the ground plane is absent in this area, because it would violate the physical isolation requirements.

The FXS module has similar functions to the FXO module (see Figure 3). In this case, the Silicon Laboratories Si3210/Si3201 chipset is used. The main difference with the FXS module is that it generates ring and "battery" DC supply voltages. Silicon Laboratories employs a clever switched-mode power supply that is modulated to generate the 200-V_{pp} sinusoidal ring voltage. Although the datasheets

*Put your creativity
to the test!*

(Sep. 20 ~ Jan. 31)

Join the Ethernet Revolution!

Circuit Cellar magazine is pleased to bring you the WIZnet iEthernet Design Contest 2007. Now you can easily add Ethernet capability to your embedded project and win fame and fortune in the process. WIZnet's W5100 hardwired TCP/IP Ethernet controller will be the key to your contest success. This 3-in-1 chip solution brings TCP/IP implementation without an OS. Both MAC and PHY are embedded.

Show us how you use the impressive W5100 to usher your embedded project into the Ethernet revolution. Your creativity and design skills could win you a share of \$15,000 in cash prizes and recognition in Circuit Cellar magazine.

**For details, visit
www.circuitcellar.com/wiznet.**

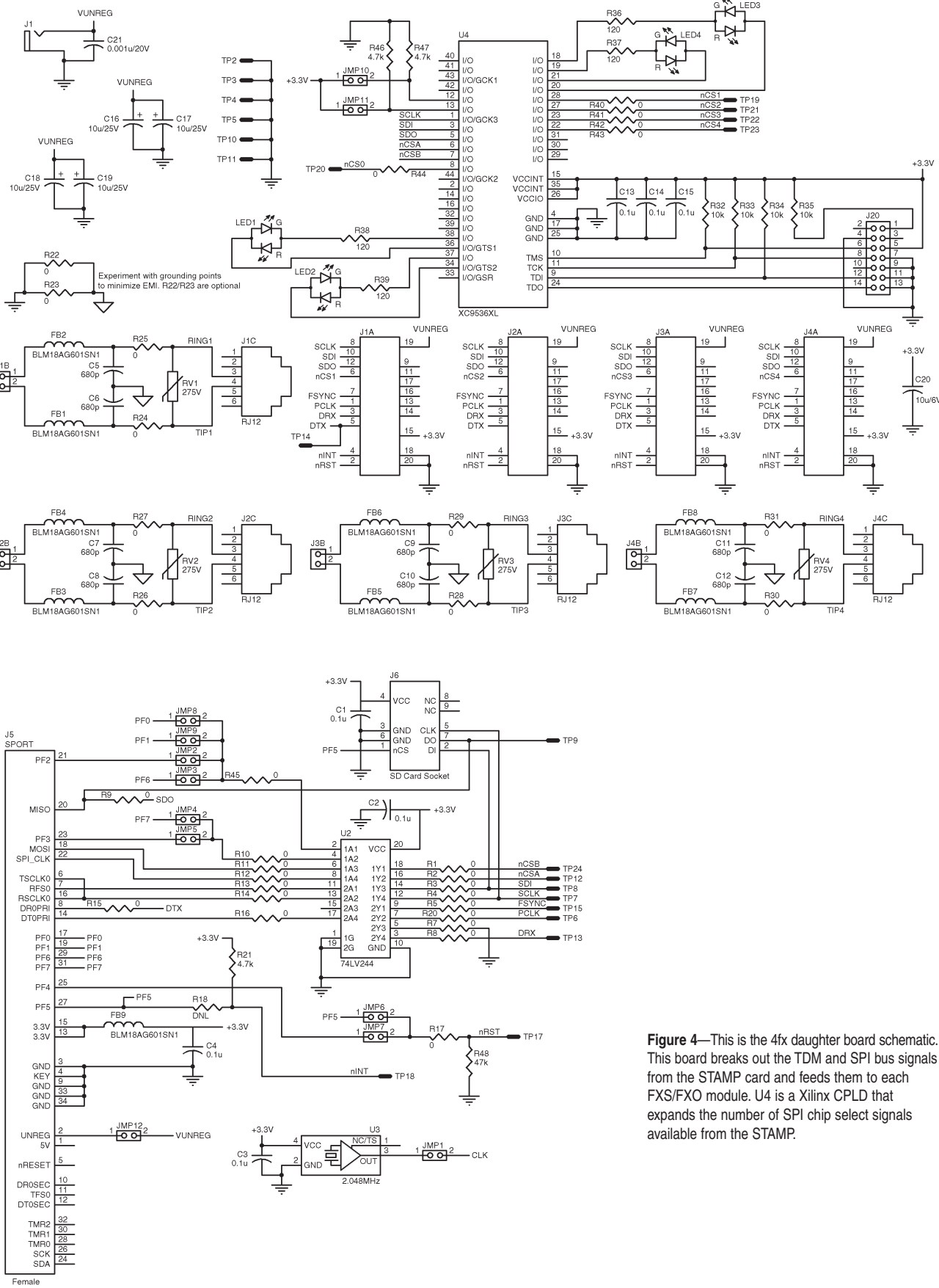


Figure 4—This is the 4x daughter board schematic. This board breaks out the TDM and SPI bus signals from the STAMP card and feeds them to each FXS/FXO module. U4 is a Xilinx CPLD that expands the number of SPI chip select signals available from the STAMP.



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

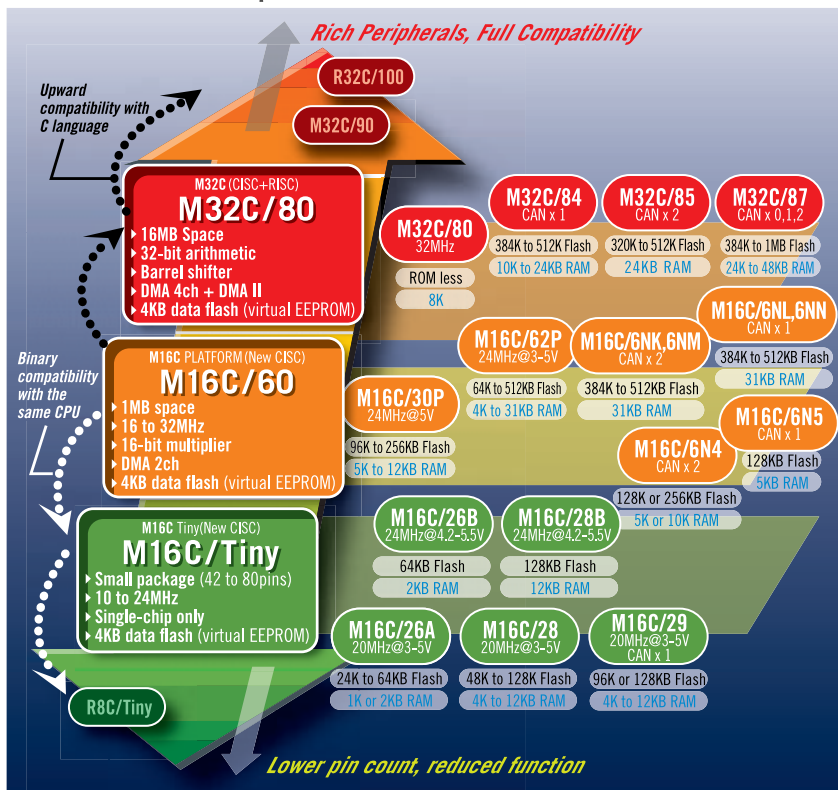
No.1* supplier of microcontrollers in the world

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.

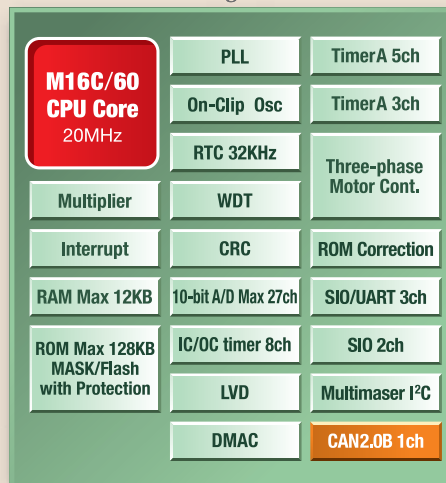
M16C Product Roadmap



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.

Source: Gartner(March 2007) "2006 Worldwide Microcontroller Vendor Revenue" GJ07168



Get Started Today -

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

www.america.renesas.com/ReachM16C/d



Renesas Technology Corp.

Everywhere you imagine. **RENESAS**

claim that the DC/DC converter can work well down to 5 V, I found that a 12-V supply was required for the DC/DC converter to avoid excessive analog noise.

The Si3210/Si3201 FXS chipset also takes care of other functions, such as line voltage monitoring, loop current detection, overload protection, impedance matching, and A/D and D/A conversion of the speech signal.

The FXS module's PCB layout is quite challenging because there are three distinct signal areas in the small 50 × 25 mm area. (A screenshot of the FXS module PCB is posted on the *Circuit Cellar* FTP site.) The first area handles line-level analog signals from the telephone handset. The second region is a digital interface containing a 2.048-MHz TDM bus and a SPI bus that connects to the fast rise time (i.e., noisy) signals from the Blackfin. The third area is a switch-mode DC/DC converter that converts a low-voltage 12-VDC rail to the -90 VDC required for the telephone "battery" supply and also generates the 200-V_{pp} ring voltage. To generate the high voltages at even small currents, large pulsed currents must flow in the 12-VDC supply line, which is a potential source of noise.

It is important to prevent the DC/DC converter and digital side from injecting noise into the sensitive low-level analog section. Based on tips from the Silicon Laboratories application notes, a few tricks were used. The DC/DC converter ground was kept isolated from the ground plane and connected only at a single point. This prevents large ground current spikes from entering the ground plane. Large current pulses in the ground plane get converted to voltages (because the ground plane impedance is small but nonzero), which then get superim-

A2	A1	A0	SPI Device
0	0	0	nCS0: spare
0	0	1	nCS1: Port 1
0	1	0	nCS2: Port 2
0	1	1	nCS3: Port 3
1	0	0	nCS4: Port 4
1	0	1	nCS5: LED register

Table 1—This truth table relates addresses to the SPI device that you are currently talking to. Note that only five devices are decoded. Four of the devices are FXS/FXO modules. Device 5 is a bank of bicolor LEDs.

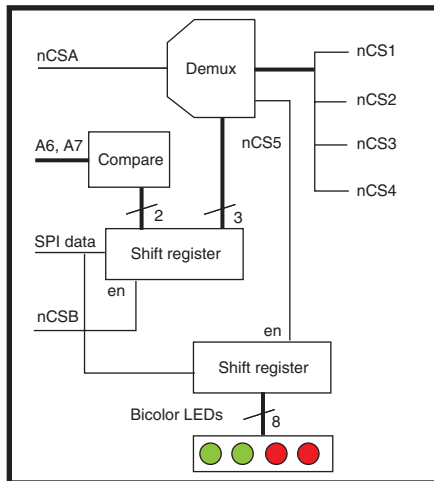


Figure 5—The CPLD expands the number of SPI chip select lines available using a demultiplexer and shift register. It also implements a SPI peripheral that drives the bicolor LEDs.

posed on the low-level analog signals as unwanted noise. You can see the lack of a ground plane in the upper-left hand part of the FXS module PCB screenshot on the *Circuit Cellar* FTP site.

A ground plane was used throughout the analog section and in the digital section. The two ground planes are connected only at a single point to prevent digital currents from flowing through the analog section and inducing noise. This point is as far away from the DC/DC converter as possible.

Bringing the two module designs to life was surprisingly straightforward. The first FXO module I built had some problems with clicks and pops in the audio; however, a good cleaning with the flux remover fixed that! The flux I use is conductive so any residue tends to upset circuits that depend on high-resistance values in certain areas.

I then tried the FXS module and it worked on the first try. I was really happy about that. I was placing calls about 5 minutes after I applied power. Hardware development isn't meant to work like that!

4fx DAUGHTER BOARD DESIGN

Now, take a look at the 4fx daughter board schematic (see Figure 4). The FXO and FXS modules communicate with the host processor via two serial buses. The signaling and control data flows via a SPI bus. The actual transmit and receive speech samples flow on a time

division multiplexed (TDM) bus.

The 4fx breaks out the SPORT connector from the STAMP board and feeds the SPI and TDM signals to each module. Because the STAMP has only a limited number of SPI chip-select signals available, a Xilinx CPLD (U4) is used to expand the number of SPI devices that can be addressed. This is described in the next section. The CPLD also supports a "stacking" architecture where several boards can be stacked on top of each other to obtain extra analog ports.

To reduce EMI, the suppression components (e.g., ferrites and capacitors) for each port were placed on the daughter board as close as possible to the RJ-11 connectors. Each digital line in the daughter board also has series termination resistors. The resistors are initially loaded as 0 Ω, but this can be increased to combat EMI or ringing issues if required. A 74LV244 buffer (U2) is also used to reduce the edge rates of high-speed digital signals from the Blackfin STAMP card. Reducing the edge rates reduces EMI because slow rise and fall times mean reduced high-frequency energy.

XILINX CPLD FIRMWARE

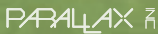
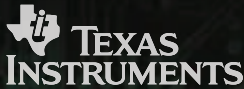
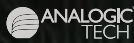
On the 4fx card, a Xilinx CPLD is used to expand the number of SPI select lines available and provide a register to drive the LEDs (see Figure 5).

The STAMP SPORT connectors have only a limited number of SPI select lines available. However, on the 4fx design, I have many SPI devices (e.g., four telephony ports and a register to drive the LEDs). I would also like to stack the 4fx cards to build eight port telephony systems. So, the problem is accessing multiple SPI devices across multiple cards using only a small number of SPI lines available on the SPORT connector.

D1	D0	LED1
0	0	Off
0	1	Red
1	0	Green
1	1	Off

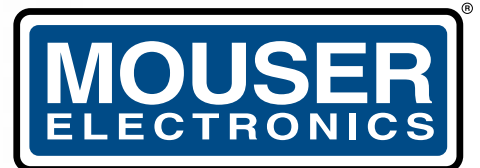
Table 2—This is a truth table for LED status. Two digital lines are used to drive each LED, which enables you to select the polarity as well as switch the LED off and on.

The Newest Semiconductors



The ONLY New Catalog Every 90 Days

Experience Mouser's time-to-market advantage with no minimums and same-day shipping of the newest products from more than 335 leading suppliers.



a tti company

The Newest Products For Your Newest Designs

(800) 346-6873



www.mouser.com

Over 860,000 Products Online

I decode a large number of SPI devices using two SPI select lines called nCSB and nCSA. nCSB is asserted and a byte is sent that selects the SPI device you want to address. The format is:

```
D7 D6 D5 D4 D3 D2 D1 D0
A7 A6 X X X A2 A1 A0
```

A [7:6] selects the card from other cards in a stack. Two jumpers determine the address of the card. Each card in the stack must have a unique 2-bit address to be successfully decoded. A [2:0] selects the SPI device on the card (only five devices are decoded) (see Table 1).

nCS [4:1] is routed via I/O pins to devices external to the CPLD. nCS5 is an internal SPI device and is not routed to an I/O pin. Once the SPI device has been selected, nCSA is asserted to talk to the actual SPI device. nCSA may be asserted as many times as required (e.g., to perform a block transfer on the SPI device).

Note that once the device has been selected, SPI transfers proceed normally (i.e., assert nCSA and read/write as desired). When access to another SPI device is required, nCSB is asserted and the byte sent can be used to select another device (or card).

The LEDs appear as an 8-bit SPI register at address five on the card. When you write to the register, the value determines the status of each

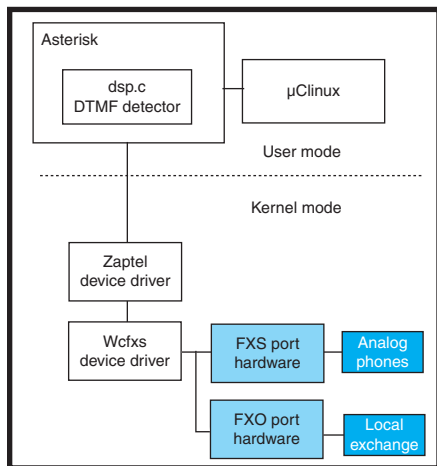


Figure 6—The IP-PBX runs the μClinux operating system and Asterisk PBX software. To run Asterisk on the Blackfin, the DTMF detector was modified to use fixed-point arithmetic. The wcfxs FXS/FXO port device driver was also modified to support the Blackfin serial port and SPI hardware.

Listing 1—Take a look at the floating-point DTMF detector inner-loop code. This is the core of a DTMF tone detector. It executes 64,000 times for every channel running on the IP-PBX. The final line implements a typical DSP operation: multiply two numbers and then add and subtract from the result.

```
static inline void goertzel_sample(goertzel_state_t *s, short sample)
{
    float v1;
    float fsamp = sample;

    v1 = s->v2;
    s->v2 = s->v3;
    s->v3 = s->fac * s->v2 - v1 + fsamp;
}
```

Listing 2—This is fixed-point DTMF detector inner loop code. The DSP operation has been broken down into two lines. Note the casting to 16-bit integers and scaling via shifts. These operations give you fine control over how the C compiler treats these operations. The main challenge with fixed-point DSP is keeping the dynamic range of signals inside a 16-bit range.

```
#define AMP_SCALE 8
#define FAC_SCALE 14

static inline void goertzel_sample(goertzel_state_t *s, short sample)
{
    int16_t v1_fix;
    int mpy;

    v1_fix = s->v2_fix;
    s->v2_fix = s->v3_fix;
    mpy = (int16_t)((int)s->fac_fix * (int)s->v2_fix) >> FAC_SCALE ;
    s->v3_fix = mpy - v1_fix + (sample>>AMP_SCALE);
}
```

LED (see Table 2). And similarly for the other LEDs:

- D[3:2] LED2
- D[5:4] LED3
- D[7:6] LED4

SOFTWARE

The Silicon Laboratories chips are commonly used with Asterisk on PCI-card-based line interface hardware. Open-source drivers for the chips exist, dramatically simplifying development.

Figure 6 shows the software architecture of the Asterisk implementation used for the embedded IP-PBX. Asterisk is a user-mode application that communicates with a kernel-mode driver called zaptel. Zaptel communicates with lower-level drivers that talk to the actual hardware.

The Asterisk wcfxs driver was chosen as the starting point for the embedded IP-PBX. The driver was originally written to communicate with the Silicon Laboratories chips operating on a PCI bus. To port the driver to the Blackfin, the PCI interface component was carefully removed and replaced with code that interfaces to

the Blackfin’s SPORT (TDM serial port) and SPI hardware.

Surprisingly, it is actually easier (and cheaper) to interface the Silicon Laboratories chips to the Blackfin compared to an x86 PC because no PCI bridge is required. The driver is also simpler.

Most of the software changes were confined to the wcfxs driver, although some changes to the Asterisk application were also required, such as architecture-specific word-alignment issues and several Linux-μClinux porting issues. One area that needed optimization was the DTMF detection routines. The code was written to run on a general-purpose x86 CPU where it is assumed floating-point support is available in the form of a hardware floating-point unit (FPU). The Blackfin does not have an FPU and is in fact optimized for fixed-point operation. Thus, the original float DTMF code ran very slowly, consuming 31 MIPS per channel.

The problem was traced to the “inner loop” of the DTMF detector (see Listing 1). The code implements a digital filter and is called eight times

for each input sample (eight filters are required for each DTMF detector). Because each channel has 8,000 input samples per second, the total number of function calls is $8 \times 8,000 \times$ the number of channels per second. So, it is important to make sure the "inner loop" code runs as efficiently as possible.

The trick is to replace the floating-point code with equivalent fixed-point code (see Listing 2). The code maps directly to the assembler instruction set of the Blackfin and compiles down very efficiently. The scale factors were chosen to keep the dynamic range of the variables within a range easily represented by a 16-bit integer. The fixed-point port was tested with a unit test from the Spandsp library that puts the DTMF detector through its paces (www.soft-switch.org).

The result was that the fixed-point DTMF detector worked just as well as the floating-point version, but it consumed about 1.75 MIPS compared to the 31 MIPS required for the floating-point version. Standard vanilla C code was used (see Listing 2). With a little Blackfin assembler replacing the C code, the performance could be improved even further. However, with 500 MIPS available on the Blackfin, 1.75 MIPS for the DTMF decoder is probably fast enough.

GO BUILD

In this article, I explored the architecture of an embedded IP-PBX and described in detail the schematic-level hardware design and PCB layout. Using embedded techniques and open-source hardware and software, it is possible to build a low-cost IP-PBX with features (VoIP, IVR, and flexibility) that rival those of \$10,000 PBXs.

Other areas of the project that have not been mentioned due to space limitations include the Asterisk software configuration, a custom DSP motherboard and configuration of the PBX. For more information, visit the project web site (www.rowetel.com/ucasterisk).

The software and hardware designs for the project are open-source and contributions by corporations or individuals are welcome. Currently, a team of companies and individuals is working on the project in areas like DSP, uClinux software, and hardware

development. Refer to the project web site for more information.

If you would like to get started with embedded IP-PBX development, I have a fully assembled and tested IP04 IP-PBX for \$450 plus shipping. ☒

Author's Note: Thanks to the Linux, Blackfin, Asterisk, and gEDA communities. My wife Rosemary did a great deal of the schematic entry for the FXS module, and Jerry Zeng from Analog Devices was very helpful in checking the design and brainstorming the CPLD requirements for the 4fx.

David Rowe has 20 years of experience in the development of DSP-based telephony and satellite communications hardware/software. He has a wide mix of skills, including software, hardware, and project management. He earned a Ph.D. in DSP Theory. David has held executive-level positions in the satellite communications industry (www.dspace.com.au) and has built and successfully exited a small business (www.voicetronix.com). However, he has decided that he is better

at debugging machines than people, so he currently hacks telephony hardware and software full time.

PROJECT FILES

To download code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/208.

RESOURCES

Blackfin Linux Project, www.blackfin.uclinux.org.

gEDA Project, www.geda.seul.org.

D. Rowe, "Free Telephony Project: Open Embedded Telephony," 2007, www.rowetel.com/ucasterisk.

S. Underwood, "Spandsp Project," www.soft-switch.org.

SOURCES

Blackfin STAMP

Analog Devices, Inc.
www.analog.com

Si3050/Si3019 and Si3210/Si3201 chipsets
Silicon Laboratories, Inc.
www.silabs.com



QuickUSB
www.quickusb.com

**With QuickUSB you can design
Hi-Speed USB 2.0 into your
product quickly without
learning USB!**



QuickUSB[®] Module
A complete USB device that you design into your product



**QuickUSB[®] Chip Pack
and iChipPack[™]**
Lets you design the QuickUSB circuit into your product and license the firmware/driver/dW for production



**QuickUSB[®] Cyclone II
Development Kit**
Combines a QuickUSB Module with an Altera CycloneII FPGA

QuickUSB Features

- * One high-speed 8/16 bit parallel port (20 Megabytes/s transfer rate)
- * Up to three 8-bit parallel ports
- * Two EIA/TIA 563 serial ports (RS-232 compatible)
- * One I2C master port
- * One SPI master port
- * One FPGA configuration port

QuickUSB Includes

- * USB driver and DLL
- * Multiple firmware files for different high-speed parallel port I/O models
- * Diagnostic utility
- * Firmware programming utility
- * Host software examples
- * User Guide & Quick Start Guide

**For a limited time, buy online at www.quickusb.com and get
20% off your order! Just enter discount code CC1107**



NimbleSig

A Compact DDS RF Signal Generator

The NimbleSig is a versatile DDS RF signal generator built around a Luminary Micro LM3S811 microcontroller. This handy system, which provides a frequency-agile RF output signal source with 1-Hz step resolution, is also capable of low-level (–50 to 10 dBm) RF power measurement.

Modern integrated circuits make the generation of stable RF signals a relatively trivial chore compared to the technology of earlier times. Typical 1960s-era VHF signal generators were relatively huge (about the size of carry-on-luggage) and somewhat impractical to use because of poor frequency stability. The manual process of tweaking the signal to within 1000 Hz of the desired frequency took some tedious fine-tuning. The low-level RF power meters of that era often needed to be renormalized to zero every few minutes.

With current technology, it is now possible to construct a shirt-pocket-size VHF/RF signal source/power meter module that maintains 10-Hz stability at room temperature for days on end and can be readily shifted in frequency in 1-Hz step increments. It takes only one additional IC to add a stable RF power meter capable of measuring levels lower than a nanowatt. The \$100 component cost for such a modern-day, dual-function module seems like a technoeconomic boon when you consider the performance and original cost of legacy RF test equipment. RF instrumentation is yet another area of technology where there have been huge improvements during the past few decades.

From the introduction of relatively low-priced direct digital synthesis (DDS) and logarithmic RF detector ICs, I have built various low-cost RF signal sources

and detectors based on new semiconductors as they have become available. During the past decade, I have witnessed a lot of improvement in DDS technology. The maximum output frequency of my first DDS generator was limited to about 20 MHz. Current technology now supports 10 times that. Recently developed DDS ICs that are capable of generating 400-MHz signals are now becoming available at more attractive costs as technology continues to evolve.

In this article, I will describe a DDS-based RF signal generator/level detection module. It is intended to be used either with a PC as a signal generator or integrated into equipment that requires an integral RF signal source. I

named the module NimbleSig for its quick-frequency agility, which enables it to act as a signal source across more than 11 octaves of spectrum (see Photo 1). The NimbleSig design uses a Fox Electronics FOX924B TCXO for a reference oscillator source, which provides good frequency stability. DDS control is provided by a 50-MHz, 32-bit Luminary Micro LM3S811 microcontroller linked to the DDS chip at a data rate of 25 Mbps, which is fast enough to accommodate many modulation needs.

DESIGN & HARDWARE

My goal for this project was to build a compact, inexpensive, VLF-to-VHF, AM/FM/CW multimode signal generator module that could service the sig-

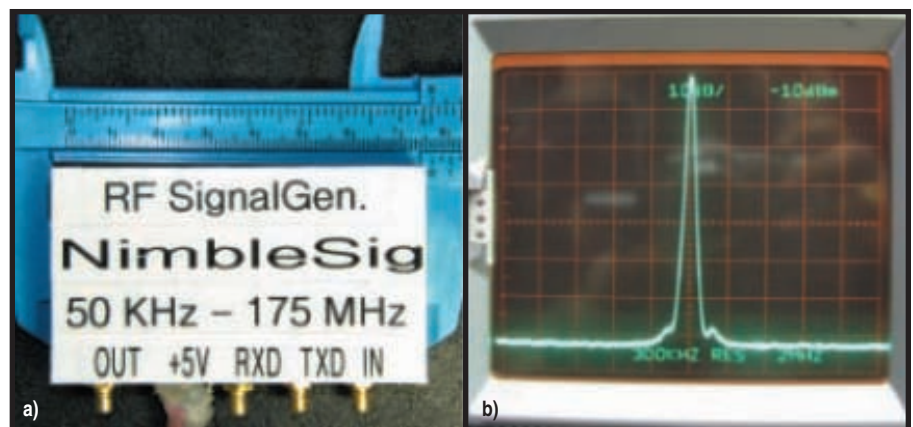


Photo 1a—This is the completed NimbleSig module. It is framed with a caliper to illustrate its small size in both English and metric units. **b**—The NimbleSig output spectrum analyzer display spanning 30 to 50 MHz. A 40-MHz full signal is at –10 dBm in the center of the screen.

nal source needs of a wide variety of applications. Figure 1 is a block diagram of the hardware. Figures 2 and 3 show the circuitry details.

The RF generator engine uses either an Analog Devices AD9859 or an AD9951 DDS IC. Both chips employ an internal 400-MHz PLL clock, which enables them to generate signals up to about 175 MHz. The compatibility of the two devices gives you a price/performance choice to best match the needs of the intended application. The AD9951 can generate a cleaner output spectrum because it uses a 14-bit DAC compared to the 10-bit DAC in the AD9859. Also, the 14-bit

resolution of the AD9951 provides finer output-level control over a wider dynamic range, which is usually desirable for a signal generator application.

processor family is well supported by Keil's μ Vision IDE, which employs ARM's C compiler. Luminary Micro's low-cost (about \$50) LM3S811 evaluation mod-

For constant output-level applications where the DDS always runs at a maximum output level, the AD9859 might be the best techno-economic choice.

The LM3S811 microcontroller uses a 32-bit ARM Cortex-3 architecture core that runs at 50 MHz, has 64 KB of program memory, and 8 KB of RAM. It also has many peripherals, including the timers, UART, ADC, and SSI blocks needed for this project. Software development for the proces-

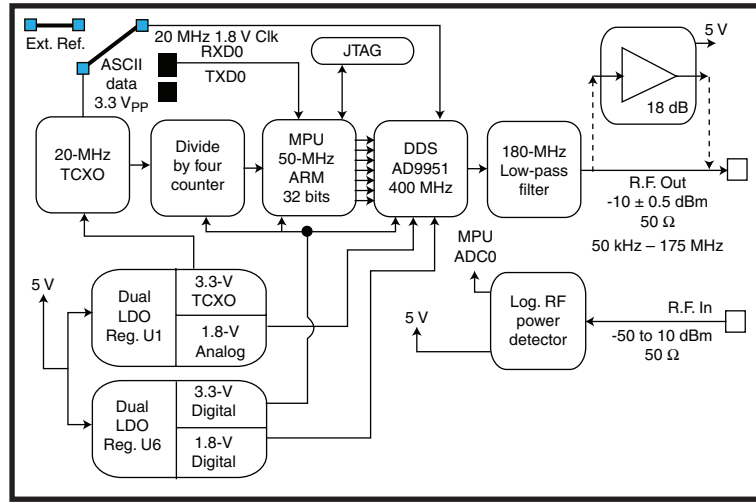


Figure 1—This is a block and level diagram of the NimbleSig design. The module is powered by a regulated 5 V. The four onboard regulators provide the voltages needed by the TCXO, divider, DDS, and MPU. The external reference injection and post power amplifier are optional.

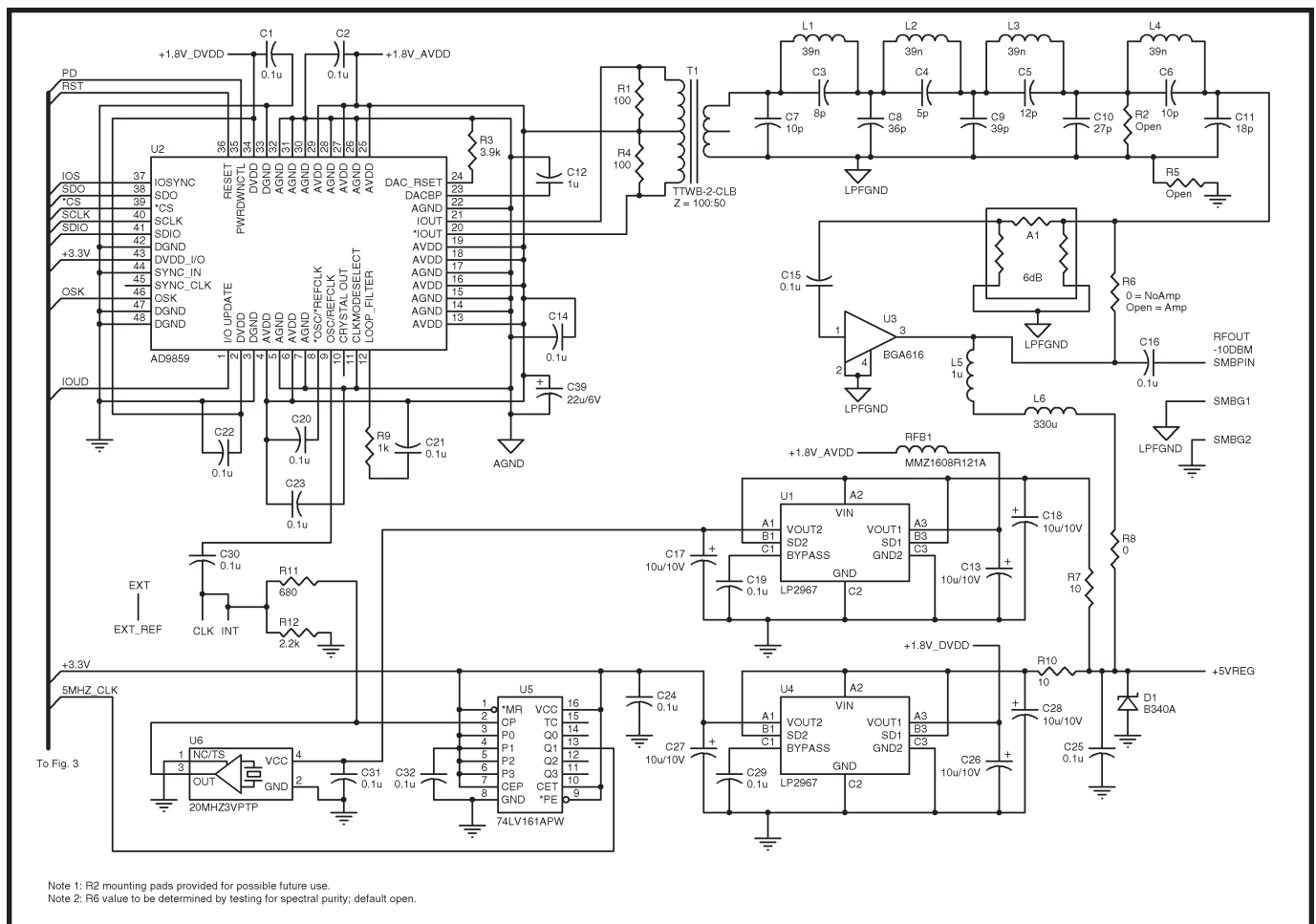


Figure 2—The NimbleSig schematic shows the DDS and analog circuitry, which includes a TCXO, a frequency divider, a DDS, 1.8-V/3.3-V regulators, an output low-pass filter, and an optional power amplifier.

ule (EVM) can be used for both MPU programming and firmware debugging. This makes setting up a software development workstation for Luminary Micro's Stellaris microcontroller family easy and inexpensive.

The small FOX924B TCXO was chosen for the 20-MHz frequency reference oscillator. The temperature stability for this device is specified at ± 2 ppm from -40° to 85°C , which should meet the needs of most applications. An external reference can be used for applications with

more demanding frequency stability and phase noise requirements.

The FOX924B circuit provides two 20-MHz output signals, one at 1.8 V_{pp} and the other at 3.3 V_{pp} . The 1.8-V signal is used to provide a clock reference for the DDS. The internal PLL of the DDS multiplies the 20-MHz clock by 20 to achieve an internal clock rate of 400 MHz. The 3.3-V_{pp} , 20-MHz output of the FOX924B feeds a low-voltage CMOS counter IC, which divides the frequency down to 5 MHz. This 5-MHz, 3-V_{pp} logic signal matches the clock-reference requirements of the LM3S811 for locking its internal 200-MHz PLL, which is internally divided by four to form the 50-MHz microcontroller system clock.

The FOX924B is powered by a dedicated 3.3-V low-noise regulator because noise on its TCXO power would tend to frequency modulate the reference clock, which would degrade the output's spectral purity. For similar reasons, a separate regulator was also provided to supply the DDS chip's analog power bus.

The proven Analog Devices AD8307 logarithmic RF power detector was chosen for the RF power level measurement device. The wonderful detector

has a 0- to 500-MHz bandwidth and offers less than ± 1 -dB tracking error from roughly -75 to 17 dBm. Compensation for the frequency response of the AD8307, which rolls off a few decibels in the VHF band, is accomplished here within the MPU's firmware.

The MPU sets the state of various DDS control inputs with 3.3-V logic from GPIO pins. The MPU's SSI peripheral provides the data communications path to the DDS chip. The SSI peripheral provides a pair of 16-word internal FIFO buffers for both output and input data. The architecture gives the MPU time to efficiently pass data in parallel format to and from the independent SSI.

The communications path to the host controller is provided by the internal UART0 peripheral. The interface passes regular 3-V unipolar data directly from the microcontroller via protection circuitry to the outside world. As the 3.3-V powered LM3S811 has 5-V tolerant I/O pins, the data ports may be directly connected to either 3- or 5-V logic external devices. Design details for my RS-232C adapter, which I use to interface to my PC serial port, can be found in the NimbleSig contest file

posted on the *Circuit Cellar* web site (www.circuitcellar.com/designstellaris2006/winners/1648.html). SMB coaxial connectors similar to the type used for the RF ports are also used for the UART, TXD, and RXD data. Coaxial connections for the data provide shielding for any potential EMI egress or ingress from or to the RF tight NimbleSig cast aluminum enclosure.

Programming and debugging the MPU is accomplished via the JTAG port connected to the LM3S811 EVM, which in turn is connected to the development workstation PC via a USB port. The EVM connection to the NimbleSig module uses a custom-made ribbon cable that adapts the JTAG standard 20-pin header to the eight-pin, lower-profile SIP socket used here. The design for this rather simple cable is also described in the contest entry.

As you can see in Figure 1, the DDS chip's output is fed into a 180-MHz low-pass filter, which blocks the 400-MHz clock and the sideband images that are always present in the raw output from a DDS DAC. For example, in the case of generating a 175-MHz signal, there would also be images present at 225 MHz

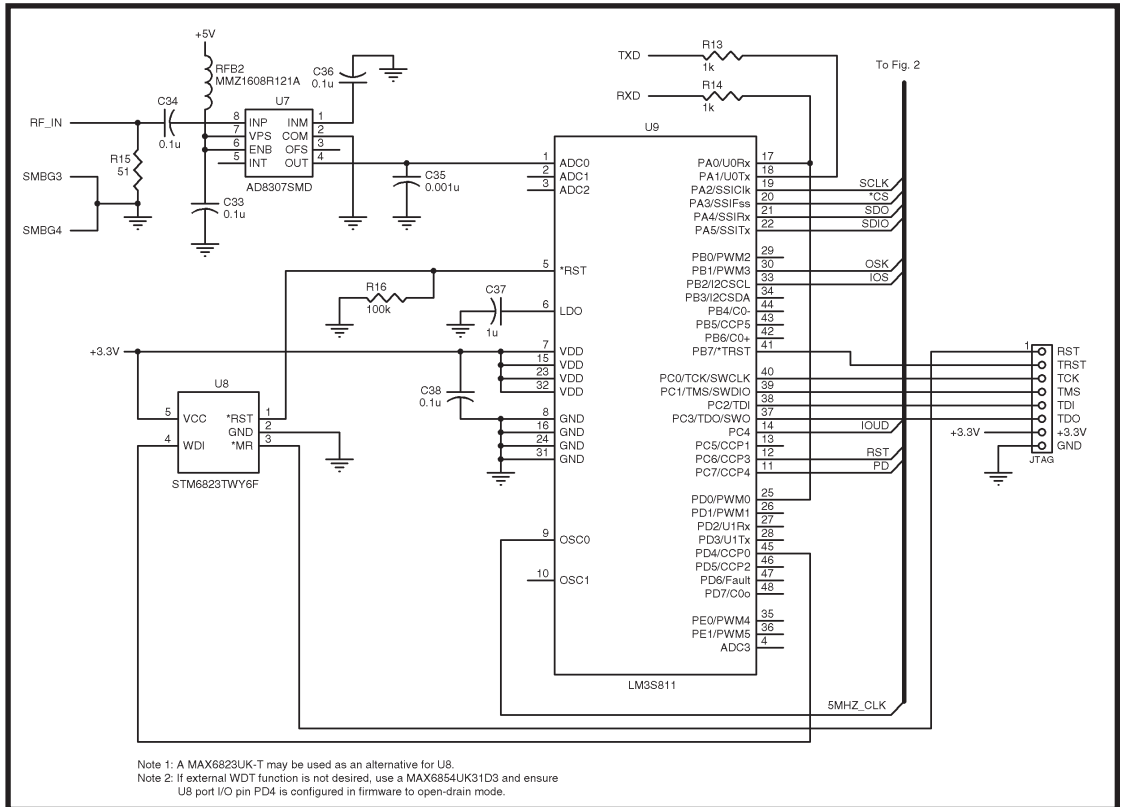


Figure 3—The NimbleSig circuitry schematic shows the MPU, MPU supervisor, power detector, serial UART port, and JTAG interfaces.

BitScope USB Mixed Signal Oscilloscope

Analog + Digital

Digital Storage Oscilloscope

- ✓ Dual Channel Digital Scope with industry standard probes or POD connected analog inputs. Fully opto-isolated.

Mixed Signal Oscilloscope

- ✓ Capture and display analog and logic signals together with sophisticated cross-triggers for precise analog/logic timing.

Multi-Band Spectrum Analyzer

- ✓ Display analog waveforms and their spectra simultaneously. Base-band or RF displays with variable bandwidth control.

Multi-Channel Logic Analyzer

- ✓ Eight logic/trigger channels with event capture to 25ns.

DSP Waveform Generator

- ✓ Built-in flash programmable DSP based function generator. Operates concurrently with waveform and logic capture.

Mixed Signal Data Recorder

- ✓ Record to disk anything BitScope can capture. Supports on-screen waveform replay and export.

User Programmable Tools and Drivers

- ✓ Use supplied drivers and interfaces to build custom test and measurement and data acquisition solutions.

Inventing the future requires a lot of test gear...

...or a BitScope



BS100U Mixed Signal Storage Scope & Analyzer

Innovations in modern electronics engineering are leading the new wave of inventions that promise clean and energy efficient technologies that will change the way we live.

It's a sophisticated world mixing digital logic, complex analog signals and high speed events. To make sense of it all you need to see exactly what's going on in real-time.

BS100U combines analog and digital capture and analysis in one cost effective test and measurement package to give you the tools you need to navigate this exciting new frontier.



Standard 1M/20pF BNC inputs



Smart POD Connector

Opto-isolated USB 2.0

12VDC with low power modes

BitScope DSO Software for Windows and Linux

BS100U includes BitScope DSO the fast and intuitive multichannel test and measurement software for your PC or notebook.

Capture deep buffer one-shots, display waveforms and spectra real-time or capture mixed signal data to disk. Comprehensive integration means you can view analog and logic signals in many different ways all at the click of a button.

The software may also 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 another PC as if a BS100U was locally connected.



www.bitscope.com

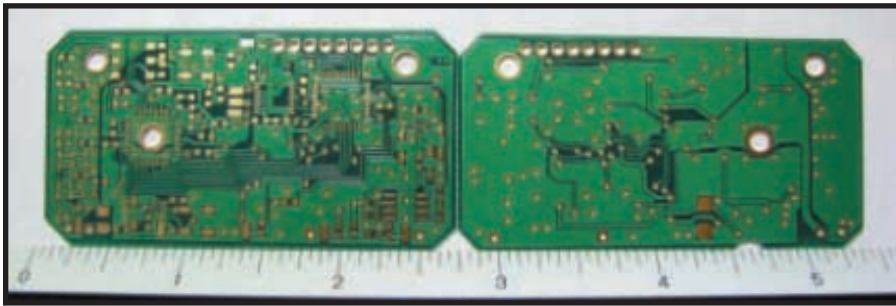


Photo 2—Here are both the component and bottom sides of the NimbleSig revision 2 bare PC board. The large center plated through hole in the center of the DDS footprint is there to enable you to solder the analog ground tab from the bottom side. The triple ground plane isolation is shown in the lower right.

(i.e., 400 – 175) and at 575 MHz (i.e., 400 + 175). The 13-pole elliptic filter design used here has a sufficiently sharp cut-off response to pass 175 MHz and reject frequencies above 225 MHz.

The filter was designed with two excellent freeware filter CAD programs. SVCfilter by Tonne Software was initially used to design a three-section, 180-MHz elliptic LPF. The fourth section was then added and optimized with Linear Technology's LTspice/SwCAD III program. It found that it was possible to use 39-nH coils for each of the four inductors, which simplifies parts procurement.

The graph of the filter response that was plotted by the LTspice/SwCAD III is posted on the *Circuit Cellar* FTP site. The cut-off is quite sharp with the first null in the reject band around 230 MHz. The attenuation at 225 MHz is about –55 dB relative to the insertion loss at 175 MHz. Thus, the 225-MHz image (i.e., 400 – 175) is well attenuated even with the DDS operating at the maximum intended frequency limit of 175 MHz.

The minuscule, surface-mount component size 0603, 39-nH inductors (made by Murata Manufacturing) used here within the filter circuit have a minimum Q specification of 40 near the 180-MHz cutoff frequency. The Q increases to about 90 at 1 GHz with self resonance at 2.8 GHz. These characteristics

make the inductors a good choice for this application.

The LPF corner frequencies of the two filters I built varied due to component tolerances. One filter limited the flattened (–10 dBm) output range to 174 MHz while the second extended it to 178 MHz.

The output from the low-pass filter may be connected directly to the output connector to provide a maximum output level of about –5 dBm or the signal can be boosted by an optional 18-dB amplifier. The optional amplifier can provide additional output power for driving passive mixers, which typically need 7-dBm local oscillator drive levels. However, for applications that require a clean spectrum, the amplifier is not a desirable

option because the harmonics it generates significantly degrade the relatively clean spectrum from the DDS.

The NimbleSig module is designed to be powered from an externally regulated 5-VDC power supply. The externally regulated 5-V bus is used directly to power the AD8037 logarithmic power detector and the optional MMIC amplifier.

Thanks to Analog Devices's low-voltage design of its DDS cores, the NimbleSig ICs do not dissipate much power. A ground tab centered under the DDS chip provides adequate heat conduction as well as a good ground connection to the PCB's analog ground plane. The DDS chip runs barely warm to the touch.

Specific DDS chip datasheets for the ICs employed here and various associated application notes on DDS-related topics are available in the RF/IF components section of the Analog Devices web site. A more detailed component-level description of the NimbleSig's circuitry is available on the *Circuit Cellar* FTP site.

ASSEMBLY

The component and bottom sides of the PC board are shown in Photo 2. The artwork is designed to an 8-mil clearance specification and the PCB stock is 1/32" FR10.

The PC board has three distinct, mutually isolated ground planes, which result in improved noise performance. The largest ground plane that extends to the left end of the board is the digital ground. The area in the center with the large plated through hole under the DDS footprint is the analog ground. The third ground plane on the right is the filter ground.

The solder pads are gold plated. Note that a silkscreen with component designations was omitted due to resolution limitations of the design software. Because the font sizes were too large, there was insufficient room on

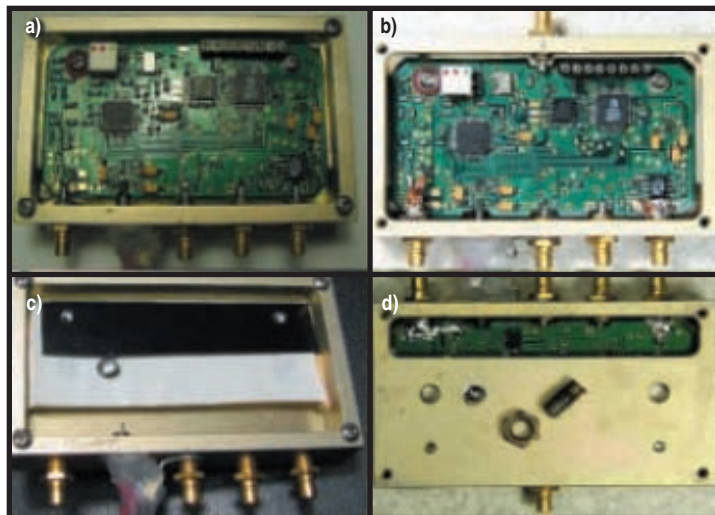


Photo 3a—The populated PCB is mounted in the chassis of the NimbleSig assembly prior to the installation of the shielding braid. **b**—The assembly has an external reference injection jack at the top and the RF shielding braid covers the center pins of the input/output jacks (jacks on the opposite bottom corners). **c**—The plastic sheet bottom isolation barriers are shown with the protruding solder bulb tip of the brass analog-ground-post screw. **d**—The brass ground post set screw has been removed, the soldered analog ground-plated through hole can be seen below the tapped screw hole, and the ground braid bonding of the triple ground planes is soldered to the threads of the gold-plated RF connectors.

the board to clearly label the components without masking the solder pads. As a better alternative, a blowup pictorial of the silkscreen layer was used for component position reference during construction.

The plated through hole in the center of the DDS footprint enabled me to solder the analog ground tab of the DDS chip from the bottom side after I finished the component-side soldering. A brass set screw threaded through a tapped hole in the chassis provided a low-impedance chassis connection to further improve analog grounding (see Photo 3).

The RF detector IC is positioned in the lower-right corner of the board to provide maximum isolation from the DDS FRF output circuitry. The detector is surrounded by ground planes on both sides of the board, which tends to reduce stray RF coupling to the detector from the rest of the circuitry. Mounting pads for the polarity protection diode can be seen on the bottom of the board near the power injection input area.

The component area on the extreme left end of the PCB is the low-pass filter area. The DDS chip is to the right of the LPF section (see Photo 3). The rows of DDS pins on the left and bottom sides (pins 25–48) mainly carry the control signals between the MPU to the IC. The reference clock is injected on the right row (pins 1–12) and the output is taken from the top row (pins 13–24). Power and ground gets applied to multiple pins in all of the rows. The two 1.8-V power buses essentially circle the perimeter of the IC.

A fiber washer is used under the left mounting screw to ensure filter ground plane isolation is not compromised. Mylar insulation barriers are placed under the PCB to ensure the electrical isolation of the bottom side of the PCB from the chassis. The three ground planes are bonded together and grounded to the chassis via the body

NimbleSig firmware – simplified flow chart

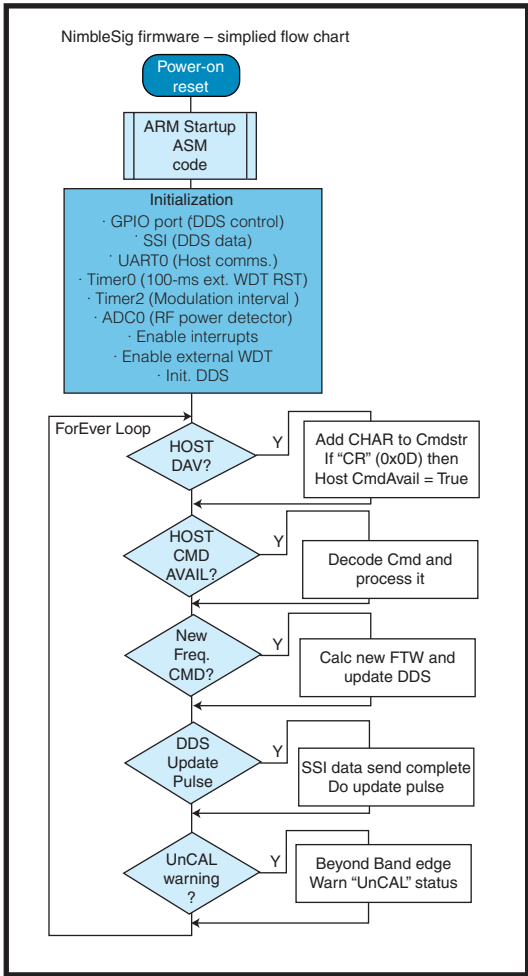


Figure 4—This is a simplified firmware flowchart of the NimbleSig firmware. The initialization routines and “forever” loop are in the main flow. On an exceptional basis, host commands are accumulated, checked, and processed. New frequency tuning words are calculated and update DDS I/O control pulses are sent when needed.

of the RF output SMB connector. A glimpse of the bonding braid can be seen in the bottom left below the corner of the PCB. The RF output and input connector center pins are partially shielded with an umbrella of braid.

The Coilcraft wideband output coupling transformer is within the white enclosure above the DDS chip. The TCXO is located to the right of the transformer. The pad for injection of an external reference is positioned along the edge to the right of the TCXO.

The JTAG connector is the eight-pin SIP socket along the edge above the MPU. Ground pin 8 is closest to the corner of the board. The 3.3-V power bus pin 7, which is not needed for the EVM, should be covered with a small piece of electrical tape. The removal of the mating pin on the ribbon cable completes a connector key.

Add SD in a FLASH

SD/MMC Serial Flash Card Library

Contents:
SD Lib Sources
Demo Projects
ECS7 Term Schematics
Bill of Material
Users manual

with our serial **Flash Card Library**

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

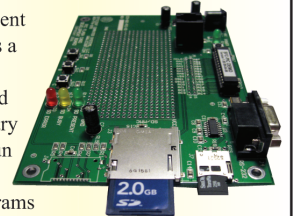
Flash Card Adapter Board

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



Development Board

The development board includes a built-in demo application and all the necessary hardware to run each of the example programs included with the SFCLIB (sold separately). It includes sockets for SD/MMC and MicroSD/TransFlash memory cards.



**EFFICIENT
COMPUTER SYSTEMS, LLC**

Offering MEGA solutions for the embedded MICRO world.

More information at: www.ECS87.com/sfclib
603-776-3151 • 5 Emerson Drive Center Barnstead, NH 03225

The divide-by-four-counter IC is located on the left of the MPU. The digital power bus dual-3.3-V/1.8-V regulator is located near the power input port below the DDS IC. The analog power bus dual 3.3-V/1.8-V regulator is located on the right end of the PCB above the Analog Devices AD8307 RF power detector IC. Although the regulators perform well, they are tiny! They are difficult to solder without an oven because of the grid array-type, leadless package.

The mounting pads for the optional

RF output amplifier chip and associated components can be seen in the bottom-left corner close to the RF output connector. Photos 3a and 3d show the grounding braid used to shield the center pins of the RF connectors. This improves the dynamic range of the RF detector and reduces the coupling of the 100-MHz synchronization clock spur into the output spectrum.

FIRMWARE

The NimbleSig firmware was written

within the restraints of the free, non-commercial, evaluation version of the Keil μ Vision IDE, which is limited to a maximum program size of 16 KB. As I write this article, the code size is about 11 KB. Thus, there is still room to add more features. However, the commercialization of the design or the expansion of the NimbleSig code beyond the 16-KB limit would require the licensing of Keil's μ Vision IDE.

The LM3S811 MPU provides sufficient onboard processing power to accurately and quickly calculate the 32-bit frequency tuning words (FTWs) with 64-bit arithmetic. The high-level C code and large program memory makes the provision of a plain-language, command-prompt-style user interface easy to implement.

The NimbleSig MPU controls the output frequency of the DDS by first calculating and then sending a 32-bit binary number called the FTW to the appropriate DDS control register. The output frequency from the DDS depends on the DDS's internal clock frequency and the value of the FTW. The FTW is calculated by the LM3S811 MPU:

$$FTW = 4,294,967,296 \times \frac{FreqO}{ClkRef}$$

FreqO is the desired output frequency and 4,294,967,296 is 2^{32} . ClkRef is the exact frequency of the reference clock, which in this case is TCXO frequency $\times 20 = 400,000,000 \pm \text{calibration}$.

The relative output level of the DDS chip is controlled by a 14-bit digital word called the amplitude scale factor (ASF). When set at $2^{14} - 1$ or 16,383 in decimal, all 14 bits are set to the one state and the full output power level is obtained from the DDS. If it is set to half the maximum value of 2^{14} (i.e., $16,384/2 = 8,192$), the carrier output voltage from the DDS is reduced to one-half the full output (-6 dB in logarithmic, relative power-level terms). Each time the number is halved, the output voltage drops by another 6 dB. NimbleSig uses the ASF register value to calibrate the output level to -10 dBm and for amplitude modulation.

In a similar manner, the phase of the output signal can be varied by up to 360° by changing the 14-bit phase

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!



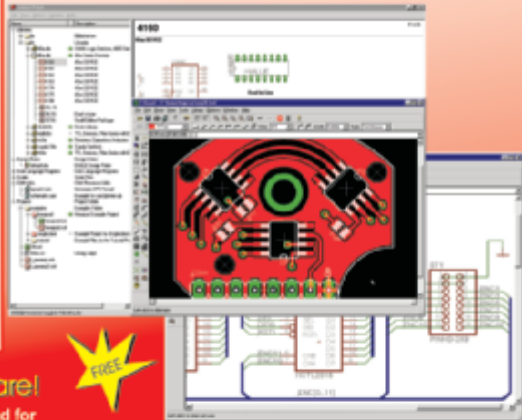
EAGLE 4.1

Schematic Capture • Board Layout
Autorouter

for Windows®
Linux®
Mac®



- 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.



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.

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

Pay the difference for Upgrades

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

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

EAGLE is a registered trademark of CadSoft Corporation. Linux is a registered trademark of Linus Torvalds. Mac is a registered trademark of Apple Computer Inc.

offset word (POW). The POW could be varied from a modulation source to phase modulate the DDS output. Details for the full DDS register set are in the ADI datasheets. Figure 4 is a flowchart of the NimbleSig firmware initialization and the main infinite loop.

Starting from the initial reset, start-up ASM code defines the stack and heap memory allocations, application interrupt vector addresses, and fault vectors. A default handler is provided for unused interrupts. Upon completion, the start-up code jumps to the entry point of the NimbleSig C firmware. Main first initializes the MPU PLL Clock mode to 50 MHz and then the various peripherals used by NimbleSig.

The general-purpose I/O (GPIO) module is enabled and pin directions are set or pin peripheral assignments are made for ports A, B, C, and D as needed for this project. UART0 is enabled for the host communications at a data rate of 115,200 bps, 8 bits, 1 stop bit, and no parity (8N1). Timer1 is initialized for triggering the ADC0 measurement of the RF level every 100 ms. ADC0 is then enabled for RF detector output-level measurement.

The SSI is initialized in the Freescale Semiconductor mode 0 format, which is compatible with the AD9859/9951 protocol. The SSI data word width is set to 8 bits. The data rate is set to 25 Mbps.

After the interrupts and external WDT are enabled in preparation for the overall operation, the initialization is completed by sending the initial control register data to the DDS via the SSI peripheral. The program then enters the main "forever loop," where it checks the status of flags. It loops here continuously except when summoned by hardware interrupts to service the needs of the peripherals. The source listings are available on the *Circuit Cellar* FTP site.

CALIBRATION

The calibration procedure depends on a well-calibrated RF power meter (e.g., a Hewlett-Packard HP432A) to establish the absolute generator output level, the utilization of the precise 10-dB level steps obtainable from the

DDS to establish the detector curve fit, the editing of constants defined in the source code, and the recompiling of the firmware. A spreadsheet, which is posted on the *Circuit Cellar* FTP site, can be used to calculate the values needed.

Photo 4 shows the modulation envelopes for different modulation frequency ranges. The envelope for frequencies less than 4 kHz looks quite classic as the 36 relatively fine modulation steps, which are hardly noticeable.

The steps become more visible as the modulation frequency increases due to the execution time of the ISR, which limits the maximum interrupt rate to about 140,000 per second. This means the maximum number of steps with a 20-kHz modulation rate is seven or less. A six-step scheme was selected for this highest range because this permits the use of the same subroutine utilized for the lower modulation ranges. Although the higher modulation frequency envelopes look pretty

\$51 For 3 PCBs
FREE Layout Software!
FREE Schematic Software!

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

expresspcb.com

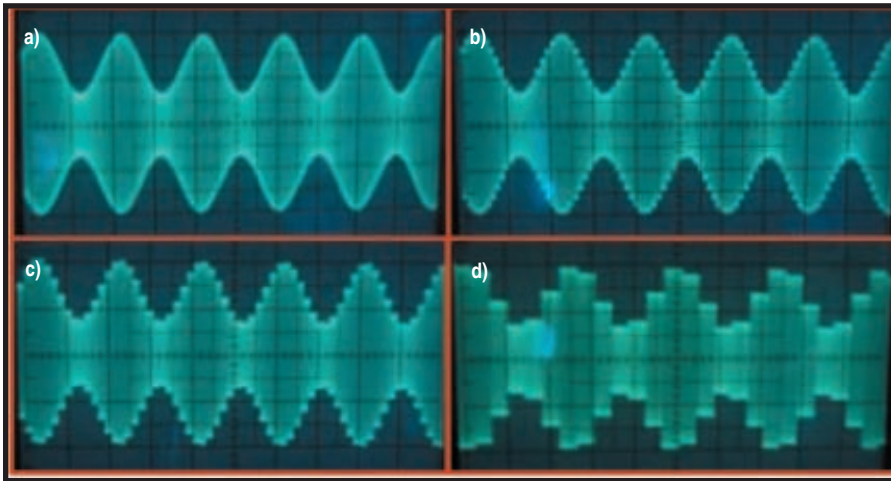


Photo 4—These are oscilloscope time domain displays of 50% amplitude modulation depth envelope photos for various modulation rates. The photos illustrate the decrease in step resolution as the modulation rate increases. The resolution is limited by the modulation ISR execution time. **a**—This illustrates the 36-modulation step resolution that is used when the modulation rate is between 1 Hz and 4 kHz. **b**—This shows 18-step resolution for between 4 and 8 kHz. **c**—Here you see twelve-step resolution for rates between 8 and 12 kHz, and the relatively coarse six-step resolution for modulation rates between 12 and 20 kHz (**d**).

rough in practice, the high-frequency ripple steps would be smoothed by the limited bandwidth of an audio receiver resulting in a restored low-distortion sine wave.

It may be possible to streamline the modulation ISR to increase the maximum interrupt rate, which would permit the use of finer modulation steps at higher modulation frequencies.

Photos 5a and 5b depict a clean spectrum down to about -70 dB. Photo 5c, which spans 0 to 1 GHz, shows the full band of interest and beyond with the generator running at full output (about -4 dBm). There is a pesky 100-MHz low-level carrier at about -57 dB and associated inter-modulation products between -55 and -60 dB down. The highest level spur at 800 MHz is probably the second harmonic of the clock, which I presume is leaking past the low-pass filter.

It should be understood that the absolute level of the 100-MHz spur and some other spurs remain fairly constant irrespective of the DDS chip output level. Thus, if the output level of the DDS is reduced via the amplitude scale factor register value or due to losses at the upper frequencies, the relative level of the spurs increases proportionally. Generally, I found that at a full output level, the spurs remained below -50 dB relative to the carrier. For the purest output spec-

trum, the DDS should be run at full output level and an external attenuator should reduce the level when needed. For many applications, these spurs are low enough to be insignificant. In my contest documentation, I describe how to shift the frequency of troublesome DDS output spurs should there be a need to do so.

OPERATING PROCEDURE

The commands, which may be listed by entering either H1 or H2 following the NimbleSig sign-on message, are listed in Table 1. The AF command sets the amplitude scale factor (ASF) register, which controls the RF output relative voltage level. For maximum output, enter AF1000000. For 50%, enter AF500000. For 10%, enter AF100000.

The B command, from B0 to B4, decreases the output level in 10-dB steps relative to maximum output. B5 turns the carrier off completely by

setting the ASF register to 0.

The reciprocal C and S commands clear or set most active bits in the CFR1 and CFR2 registers, respectively. For example, the command C112 clears CFR1 bit 12, while the command S112 sets CFR1 bit 12. The changing of any bits that could disrupt control of the DDS is prevented and a "Cmd Error!" message is echoed.

New frequencies can be entered in hertz, kilohertz, or megahertz with the FH, FK, and FM commands. To set the frequency to 123,456,789 Hz, key in the following at the ">" prompt: FH123456789<enter>. To set the frequency to 150 MHz, simply key in: FM150<enter>.

MA and MF turn on AM and FM modulation, respectively, in accordance with the settings of the MD, MP (as appropriate), and MQ commands.

R permits the reading of the DDS registers. SP permits the setting of the phase offset in millidegrees. Z can be used to put the MPU to sleep.

PROJECT COMPLETE

In general, the NimbleSig design has met my objectives. I found that Keil's IDE and Luminary Micro's EVM worked well together. Luminary Micro's API library for its LM3S811 greatly simplified the writing of the firmware. I think it should be quite easy for others to pick up where I left off to customize the firmware to a specific application.

My only regret was my selection of a power regulator in a package that was difficult to solder without an oven. If there is sufficient interest demonstrated in NimbleSig, I may redesign the PCB artwork for power regulator chips in packages that are easier to work with.

I did my best to make the overall

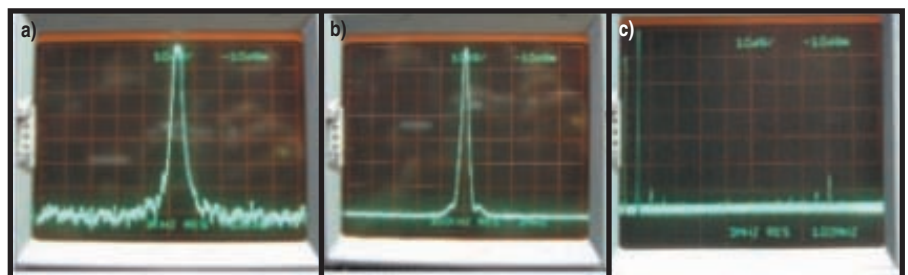


Photo 5—These are frequency domain spectrum analyzer displays of the NimbleSig generator output with a 40-MHz signal. Various spans are shown (from left to right): a 100-kHz span, a 20-MHz span, and a 0 to 1 GHz spectrum.



HI-TECH C[®] PRO for the PIC10/12/16 MCU Family

ANSI C Compiler comes with Omniscient Code Generation™

DENSER CODE in 1/2 the time

HI-TECH C PRO for the PIC10/12/16 MCU Family - Based on the reliable PICC™ STD compiler, and supporting Microchip's PIC10/12/14/16/17 series of MCUs, the new **HI-TECH C PRO for the PIC10/12/16 MCU Family** ANSI C Compiler comes with Omniscient Code Generation.

OMNISCIENT CODE GENERATION

Extracts information from multiple source files simultaneously, allowing more intelligent state-of-the-art code generation that:

- Automatically handles memory banking without requiring special qualifiers;
- 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;
- Automatically analyzes user assembly and object code files; and
- Is simple to use.

DENSER CODE, IN HALF THE TIME.

Code compiled with **HI-TECH C PRO for the PIC10/12/16 MCU Family** can deliver up to 30% denser code than you are currently producing. Additionally, automatic memory and stack management could cut your development time in half.

FREE 45 DAY EVALUATION

A fully functional 45 day trial version of the **HI-TECH C PRO for the PIC10/12/16 MCU Family** will be available, free of charge from HI-TECH's website. Order your demo at www.htsoft.com/portal/ccpicpro.

PRICE AND AVAILABILITY

HI-TECH C PRO for the PIC10/12/16 MCU Family includes, at no extra cost: HI-TECH PRIORITY ACCESS™ - 12 months access to updates and technical support; and HI-TECH SATISFACTION GUARANTEE - a hassle-free 30 day money back guarantee.

COMING SOON



HI-TECH Software proudly supports the Microchip brand with industrial-strength software development tools and C compilers. PICC is licensed exclusively to HI-TECH Software by Microchip Technology Inc.

HI-TECH
S O F T W A R E

Ph: 800 735 5715 Web: www.microchip.htsoft.com

Pololu
Robotics & Electronics

Robot Kits
Line followers
Robot arms
Hexapods
Chassis

Mechanical Components
Gearboxes, servos
Wheels, ball casters

Motion Control
Servo controllers
Motor controllers

Robot Controllers

voltage regulator
power LED (green)
red user LED (on PD1)
ATmega48/168 microcontroller
20 MHz clock
dual H-bridge
trimmer pot (on ADC7)
programming connector

Solder Paste Stencils

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

From \$25

Custom Laser Cutting

From \$35

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

1-877-7-POLOLU
www.pololu.com
6000 S. Eastern Ave. 12D, Las Vegas, NV 89119

Command	Function	Command	Function
AF1..1000000	Set amplitude scale factor	MA	AM modulation on
B0	Set RF out level to maximum	MD1..100000	FM peak deviation Hz
B1	Set level to (max. -10 dB)	MF	FM modulation on
B2	Set level to (max. -20 dB)	MP1..99	AM modulation depth %
B3	Set level to (max. -30 dB)	MQ1..20000	Modulation frequency Hz
B4	Set level to (max. -40 dB)	MX	Modulation off
B5	Turn RF output off	R1	Read & disp. CFR1 in hexidec.
C1 0..31	Clear bit in CFR1 register	R2	Read & disp. CFR2 in hexidec.
C2 0..23	Clear bit in CFR1 register	RA	Read & disp. amp scale factor
FH1..192000000	Freq = # hertz at -10 dBm	RP	Read & disp. PhaseOffset mDeg.
FK1..192000	Freq = # hertz at -10 dBm	RW	Read & disp. freq. tune word
FM1..192	Freq = # hertz at -10 dBm	S1 0..31	Set bit 00..31 in CFR1 register
H1	Display help screen 1	S2 0..23	Set bit 00..23 in CFR2 register
H2	Display help screen 1	SP0..360000	Set phase offset to millidegrees
L	Measure RF input level	Z	MPU deep sleep-wake any key

Table 1—The NimbleSig commands are all in standard ASCII, 115 kbps, and 8N1 with no flow control. Frequencies may be abbreviated in megahertz, kilohertz, or entered directly in hertz. For example, 100 kHz can be entered as “FK100<Enter>.” ASF is in μ factors to obtain maximum output send “AF1000000<Enter>,” or to obtain 50% of maximum voltage level send “AF500000<Enter>.” Similarly, phase offset is set in millidegrees.

content of this article as factual as possible; however, there are no guarantees implied about anything presented here. Thus, anyone who uses my NimbleSig design or any portion of it must accept all of the associated risks. ☒

Author's Note: A great deal of the credit for the NimbleSig belongs to Dr. James Koehler, Ph.D., who introduced me to surface-mount construction, embedded C, and ARM processors. He spearheaded the design of earlier DDS generators that we mutually developed. Without Jim's positive influence and generosity, I would not have acquired the knowledge and ability to complete the project. I would also like to thank another good friend, Mr. Graig Pearen, who helped me obtain instrumentation for my lab. Last, but certainly not least, I wish to thank my wife Sylvia who gracefully puts up with my absenteeism while I work on my projects.

Thomas Allread (nimblesig@telus.net) graduated from the CREI's Telecommunications Engineering Technology program and received certification from MANSCETT. He has worked in the telecommunications industry as a technician, an instructor, and a transmission standards engineering specialist. Subsequently, he worked for Bell Canada as a long-distance facilities management advisor in Saudi Arabia. Now retired,

Thomas lives with his wife on picturesque Vancouver Island, where he spends most of his time pursuing interests such as designing surface-mount electronics, hobby farming, RVing, digital photography, and amateur radio.

PROJECT FILES

To download code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/208.

RESOURCE

Analog Devices, Inc., “A Technical Tutorial on Digital Synthesis,” 1999, www.analog.com/UploadedFiles/Tutorials/450968421DDS_Tutorial_rev12-2-99.pdf.

SOURCES

AD8307 Power detector, AD9859 DDS, and AD9951 DDS
Analog Devices, Inc.
www.analog.com

FOX924B TCXO
Fox Electronics
www.foxonline.com

LTspice/SwCAD III Program
Linear Technology Corp.
www.linear.com

LM3S811 Microcontroller
Luminary Micro, Inc.
www.luminarymicro.com

SVCfilter
Tonne Software
www.tonnesoftware.com

We're semi nuts

We've got semis on the brain. Jameco offers more major brands of semiconductors than anyone — almost twice as many as these catalog distributors.* It's another Jameco advantage.

	Mouser®	Newark®	Allied®	Jameco®
5	Atmel Semiconductor Avago Technologies Cypress Semiconductor Diodes, Inc. Fairchild Semiconductor	Altera Analog Devices Avago Technologies Cypress Semiconductor Fairchild Semiconductor	Analog Devices Atmel Semiconductor Avago Technologies Freescale Semiconductor Infineon Technologies	Altera Analog Devices Atmel Semiconductor Avago Technologies Cypress Semiconductor
10	Freescale Semiconductor Intersil Lattice Semiconductor Lite-On Semiconductor NEC Corporation	Freescale Semiconductor Integrated Devices Intel Corporation Intersil Lattice Semiconductor	Integrated Devices Intel Corporation Intersil Lattice Semiconductor Maxim	Diodes, Inc. Fairchild Semiconductor Freescale Semiconductor Infineon Technologies Integrated Devices
15	Sharp Microelectronics ST Microelectronics Texas Instruments	Maxim National Semiconductor ST Microelectronics Texas Instruments	National Semiconductor NXP (formerly Philips) ST Microelectronics Texas Instruments	Intel Corporation Intersil Lattice Semiconductor Linear Technology Lite-On Semiconductor
20				Maxim Micron Technology Microsemi National Semiconductor NEC Corporation
25				NXP (formerly Philips) Renesas Technology Sharp Microelectronics ST Microelectronics Texas Instruments Toshiba



Free shipping
on these and
79 other brands.
Call for details.

OTHER JAMECO ADVANTAGES:

- More major passive, interconnect and electro-mechanical brands than other distributors.
- **99%** of catalog products ship the same day.
- Lowest prices guaranteed, or we pay 10%.
- Major brand names **and** generic equivalents for even greater cost savings.

JAMECO[®]
ELECTRONICS

Order 24 hours a day, 7 days a week

www.Jameco.com

Or call 800-831-4242 anytime



iEthernet Bootcamp

Get Started with the W5100

Are you ready to join the Ethernet revolution? If so, it's time to start working with WIZnet's W5100 hardwired TCP/IP embedded Ethernet controller. In this article, Fred helps you get started on your first W5100-based design.

I recently received an e-mail from a reader asking why there were no in-depth TCP/IP stack “how-to” articles. Honestly, I had never given that much thought because I normally forego the formal TCP/IP stack in favor of small, easy-to-follow, home-brewed Ethernet driver packages. As a magazine writer, my first guess on the lack of TCP/IP stack magazine literature is the cost versus interest factor. I have reviewed many commercial TCP/IP stack products and I can say from experience that you get what you pay for. My readers simply can't afford or financially justify a full-blown commercial TCP/IP stack for their applications and projects. Thus, why should I ask them to read about a TCP/IP product that they can't afford to use? My second stab at why TCP/IP stacks aren't in magazine vogue is complexity. Many of you have written articles for magazines and you know that you are limited to so many words per article. It would take a series of articles to explain everything you would need to know about TCP/IP stacks.

I must admit that in the past I have offered up some pretty pricey stuff in my articles. These days, I tend to shy away from super-expensive and complex subjects for the reasons I just outlined. However, when I see something that may be what typical technical magazine readers like you and I are looking for, I'm all over it. For instance, I have found that Ethernet ICs supported by free TCP/IP stacks are very popular with *Circuit Cellar* readers. I've also discovered that many readers who implement single-IC

Ethernet devices don't even use a TCP/IP stack. Instead, like me, they employ simple protocol drivers specifically written for the single-IC Ethernet device that they are deploying in their project. I practice what I preach, and what I'm about to introduce to you is the best of both the garage Ethernet driver and TCP/IP stack worlds. How would you like to solder down a single-IC Ethernet solution that provides the power of a full-blown commercial TCP/IP stack as if it were a set of simple Ethernet drivers? Read on, my friend.

WIZnet W5100

The WIZnet W5100 is a single-IC Ethernet solution with a built-in TCP/IP stack. The W5100 folks like to call their on-chip stack a “hard-wired stack” because all of the W5100's Internet-enabling goodies are contained within a compact 80-pin LQFP. In addition to the W5100's hard-wired TCP/IP stack, other W5100 Ethernet goodies include an integrated IEEE 802.3 10Base-T and 802.3u 100Base-TX-compliant MAC and PHY. As you would expect, the W5100's TCP/IP stack supports all of the things you need to put an embedded Ethernet gadget on a network. The W5100's TCP/IP stack supports TCP, UDP, ICMP, and ARP, which normally provide enough protocol power for

a major portion of embedded Ethernet LAN and Internet projects that are launched by folks like you and me. PPPoE is also supported by the W5100. The inclusion of PPPoE enables you to use the W5100 in ADSL applications.

If you've ever toyed with embedded Ethernet, you know that the lack of a transmit or receive buffer memory can be painful and hamper the performance of your embedded Ethernet device. The embedded Ethernet IC manufacturers are aware of this. Most of the single-IC Ethernet solutions offered these days include a fair amount of dedicated transmit and receive buffer memory. The W5100 is no exception, and it is equipped with 16 KB of internal transmit/receive buffer memory.

To avoid the exclusion of smaller microcontrollers, the W5100 can communicate with a host of microcontrollers using an SPI, direct memory access, or indirect memory access. To further accommodate the majority of today's newer microcontrollers, the W5100 is

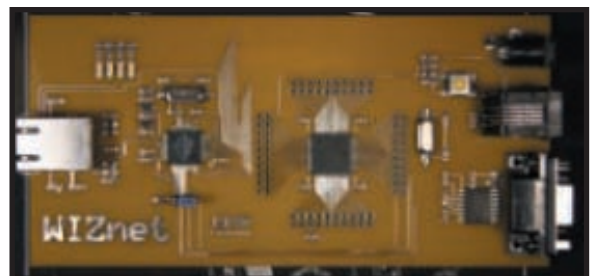


Photo 1—My WIZnet W5100 development board is based on the Microchip Technology PIC18LF8722. The PIC18LF8722 is hefty enough to enable the selective use of Direct Memory mode, Indirect Memory mode, and SPI mode access to the W5100's registers and buffer memory. Using the Microchip PIC18LF8722 also puts the powerful set of Microchip development tools at our disposal.

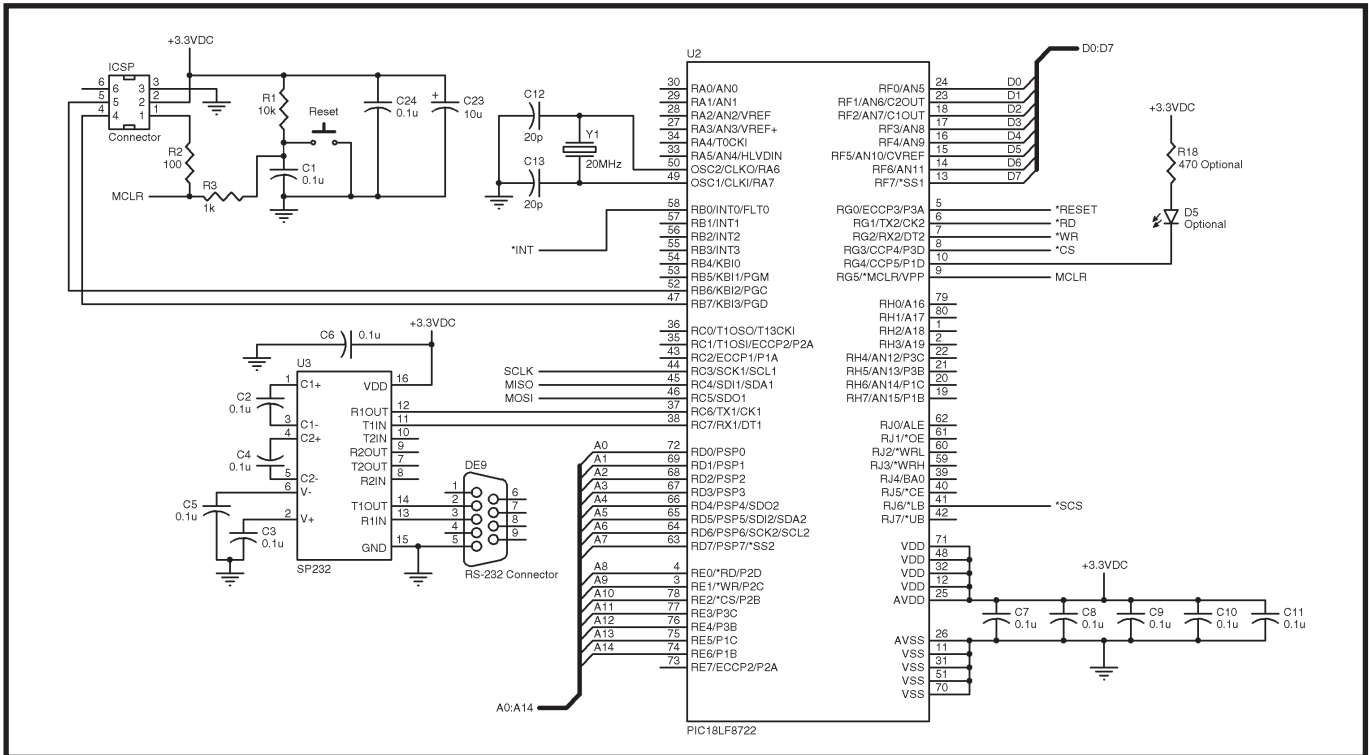


Figure 1—Nothing much I need to say about what you see here. However, the PIC18LF8722 does remind me of my favorite Military Channel quote: "It's just a good, solid tank."

powered with a 3.3-VDC power source. This enables the W5100 to be directly interfaced to low-power microcontrollers that also run on a 3.3-VDC power rail. The W5100 can also be integrated into legacy 5-VDC systems because its I/O subsystem is 5-V tolerant.

The W5100 supports up to four simultaneously active sockets. Thus, all you need to know is basic socket programming because you will be shielded from the W5100's internal Ethernet engine operations. The W5100 is designed to provide the bulk of everything needed to produce a working embedded Ethernet device while being easy to use. The only things the W5100 won't do for you are write its own code and handle IP fragmentation.

I just happen to have a couple of W5100 ICs. Let's assemble a W5100-based device from scratch. Once we've got the W5100 hardware realized, we'll put together some Microchip Technology PIC18LF8722 driver code for our W5100 development board.

BUILD A DEVELOPMENT BOARD

For your convenience, I am supplying the PCB layout for an EDTP Electronics-designed W5100 device (see Photo 1). The PCB layout file on the *Circuit*

Cellar FTP site is in ExpressPCB format. I chose ExpressPCB because it is a relatively inexpensive PCB manufacturing service that is available to everyone. ExpressPCB software is free for download, and the quality of ExpressPCB PCBs is excellent. Another plus associated with using ExpressPCB is that you don't have to design your W5100 PCB from scratch. You can use my ExpressPCB PCB template and modify it to meet your needs. If you already have a favorite PCB CAD program, you can easily port my design to your CAD format using my original drawing as a guide. As you would expect, I haven't done anything to complicate the W5100 project board design.

The EDTP WIZnet W5100 project board is basically a standard PIC18LF8722 configuration that is wired into a basic W5100 configuration. As you can see in Figure 1, the PIC18LF8722 has enough I/O to wire-in the 15-bit W5100 address bus, the 8-bit W5100 data bus, and all of the W5100 control signals (*RD, *WR, *CS, and *INT) with I/O to spare. In addition to wiring in the W5100 in Direct Bus Interface mode (A0:A14 with D0:D7 and control signals), I attached the W5100's SPI portal and an SPI select pin to the PIC18LF8722's SPI I/O inter-

face, which enables you to access the W5100's internals in W5100 SPI mode. Because the W5100's address lines are all pulled down internally, the Indirect Bus Interface mode of operation, which uses only two of the 15 address lines, all of the eight data lines, and all of the control signals can also be easily implemented with the EDTP WIZnet W5100 design.

All of the PIC18LF8722's 80 I/O and power lines are pinned out in blocks of 20 pins to standard 0.1" header pads. The PIC18LF8722 is supported by a 20-MHz clock, a Microchip-certified ICSP programming/debugging portal, and a regulation RS-232 port. I did not include any power supply circuitry because a Digi-Key-supplied 3.3-VDC wall wart does a great job powering the W5100 project board and the external programming/debugging hardware.

On the W5100 side of the EDTP W5100 development board, the W5100 is supported by the required 25-MHz crystal and an all-in-one can of magnetics (see Figure 2). I chose to incorporate the U.D. Electronic RDI-125BAG1A pulse transformer for a couple of reasons. First, the RDI-125BAG1A footprint fits exactly into the old packet whacker pulse transformer footprint, for which I

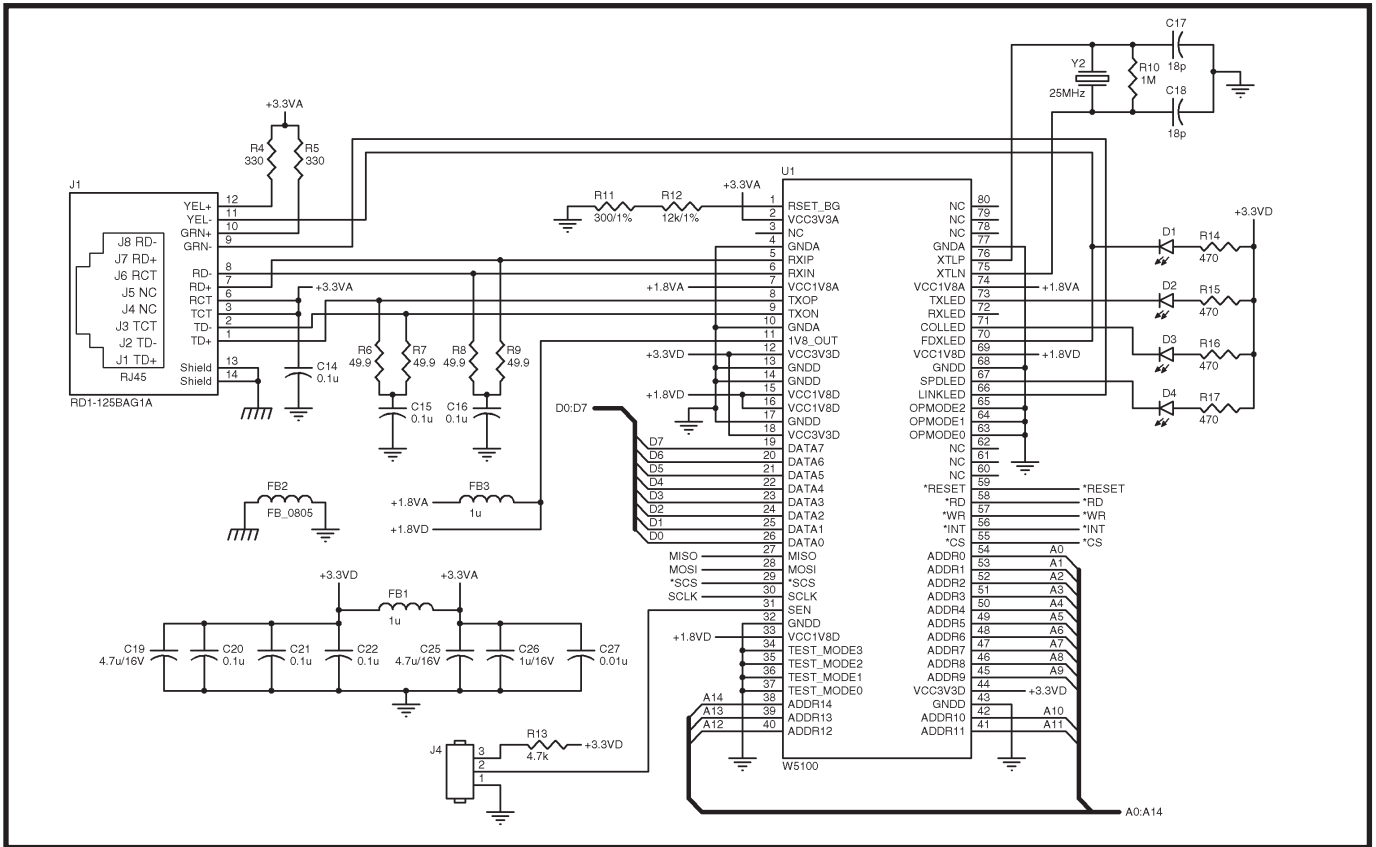


Figure 2—You can get your hands on most everything here from Digi-Key or Mouser Electronics. My friends at Saelig supply the W5100 IC. Saelig doesn't stock the RD1-125BAG1A on its site, but you can probably get the pulse transformer from many of the vendors listed on WIZnet's web site.

already have a time-proven ExpressPCB pad layout. Second, like the old packet whacker mag jack package, the RDI-125BAG1A has a pair of built-in indicator LEDs in addition to a pair of transmit and receive pulse transformers and the required internal terminating resistors. If you've ever worked with the EDTP ASIX-based and Microchip-based Ethernet development boards, you'll notice that the W5100 PHY connections are very similar to the EDTP Electronics ASIX and Microchip ENC29J60 designs.

You may wonder why there are no bypass components on the W5100's internally generated 1.8-VDC supply. That question was posted on the W5100 online technical support question-and-answer board. The W5100 engineering answer was to follow the path that was set forth by the W5100 reference schematic, which is void of 1.8-VDC supply bypass components.

Because the EDTP WIZnet W5100 project board is designed to help you get your W5100 design up and running quickly, I attached all of the W5100 LED indicator lines to LEDs. The pair of RDI-

125BAG1A LEDs is connected to the W5100's LINKLED and RXLED status indicator I/O pins. I pulled the TXLED, COLLED, FDXLED, and SPDLED indicators out to discrete LEDs, which you can see in Photo 1 hanging above the

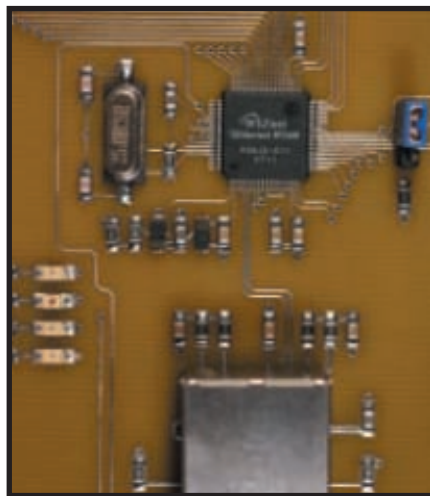


Photo 2—There's nothing here you can't handle. With the exception of the 12.3-kΩ resistor pair, the line of components closest to the W5100 is all filter and bypass components. The PHY components are in the line closest to the pulse transformer. Note the status LEDs and the SPI select jumper at the port and starboard extremes of this photo.

city of WIZnet W5100 0805 supporting SMT components. I have also provided a jumper to select W5100 SPI mode if you choose to run your W5100 in that manner. The only oddity I need to point out is the 12.3-kΩ reset resistor pair you see in Figure 2, which is attached to the W5100's RSET_BG pin. A bird's-eye view of the W5100 portion of the EDTP WIZnet W5100 development board is in Photo 2.

As you can see, the W5100 hardware is a no-brainer. Before we move on to do some W5100 coding, what you don't see in Photo 1 is the heartbeat LED I attached to RG4 on the PIC18LF8722. It's just there as a warm fuzzy to let me know that things are moving on the firmware side. I flash the RG4 LED at a rate of 1 Hz via the PIC18LF8722's Timer3 interrupt-driven real-time clock code.

WIZnet W5100 GARAGE CODE

From a WIZnet W5100 programmer's point of view, the W5100 consists of Common registers, Socket registers, TX memory, and RX memory. The W5100's

Common registers consist mostly of W5100 local IP and MAC addressing fields. Also included within the confines of the Common registers are RX and TX memory sizes and PPP/PPPoE parameters. It looks like we will be populating most of the Common registers. So, let's kill two birds with one stone and use the Common registers to test the PIC18LF8722 driver hardware by writing some basic PIC18LF8722 routines to read and write the W5100's registers. I'll use the HI-TECH PICC-18 C compiler in conjunction with MPLAB and a Microchip Technology REAL ICE as my W5100 firmware brewing tools.

About 18 hours later, I returned to write this sentence. I could not get my

W5100 to communicate correctly with the PIC18LF8722 to save my life. A cursory look at the W5100 project board didn't indicate any problems. So, I turned to my C code to see if I could find the bug. As it turns out, my C was fine, but my eyes deceived me. A great number of the PIC18LF8722 W5100 address and data I/O pins simply did not get soldered to the W5100 PCB. I use an industrial hot air reflow machine to mount fine-pitched ICs like the W5100 on a regular basis. I've done so many of them that I take the process for granted. Well, this time the reflow machine bit me.

In the meantime, I managed to put some W5100 I/O code together. The official factory W5100 driver code I have is

Listing 1—You won't find this level of coding in the W5100 datasheet examples. Nothing will whizz about without these base register I/O routines.

```
char gwayipaddrc[4] = {192,168,0,1};
char svrmacaddrc[6];

#define make8(var,offset) ((unsigned int)var >> (offset * 8)) & 0x00FF
#define TO_WIZ          TRISF = 0x00
#define FROM_WIZ       TRISF = 0xFF

*****
void wr_wiz_addr(unsigned int addr)
{
    addr_hi = (make8(addr,1));
    addr_lo = addr & 0x00FF;
}
void wr_wiz_reg(char reg_data,unsigned int reg_addr)
{
    TO_WIZ;
    wr_wiz_addr(reg_addr);
    data_out = reg_data;
    clr_WR;
    NOP();
    set_WR;
    FROM_WIZ;
}
char rd_wiz_reg(unsigned int reg_addr)
{
    char data;
    wr_wiz_addr(reg_addr);
    clr_RD;
    NOP();
    data = data_in;
    set_RD;
    return(data);
}
*****
clr_RSET;
msecs_timer2 = 0;
while(msecs_timer2 < 2);
set_RSET;
addri = GAR0;
for(i8=0;i8<4;++i8)
    wr_wz_reg(gwayipaddrc[i8],addri++);
addri = GAR0;
for(i8=0;i8<4;++i8)
    svrmacaddrc[i8] = rd_wz_reg(addri++);
```

PRESTO

The First USB In-Circuit Multipurpose Programmer



Very fast and flexible in-system programmer for many devices:

- Atmel AVR, 8051 architecture
- Microchip PIC, dsPIC, rPIC, ...
- Texas Instruments MSP430
- ARM MCUs by Atmel, NXP (Philips) - including debugging
- FPGAs, CPLDs & SCPs by Xilinx, Altera, ...
- Serial Flash, EEPROMs, ...

SIGMA

Unique Logic Analyzer With Extremely Large Memory for more than 14,000,000 events, up to 16 inputs, up to 200 MHz sampling, flexible triggering, ...

...and more

MU Beta - unique PIC emulator
USB modules - parallel & serial
SPINET - Ethernet-SPI interface
 Development boards and kits, ...

For developers by developers.

ASIX ASIX, Prague, Czech Rep.
 info@asix-tools.com

www.asix-tools.com

Advertising Calendar

January Issue #210
 Embedded Applications
 Space Close: Nov. 13

BONUS DISTRIBUTIONS:
 DesignCon, MSC

February #211
 Wireless Communications
 Space Close: Dec. 11

BONUS DISTRIBUTIONS:
 APEC; IPC Printed Circuits Expo;
 APEX & The Designers Summit; CTIA

Call: Shannon Barraclough
 (860) 872-3064
 Shannon@circuitcellar.com

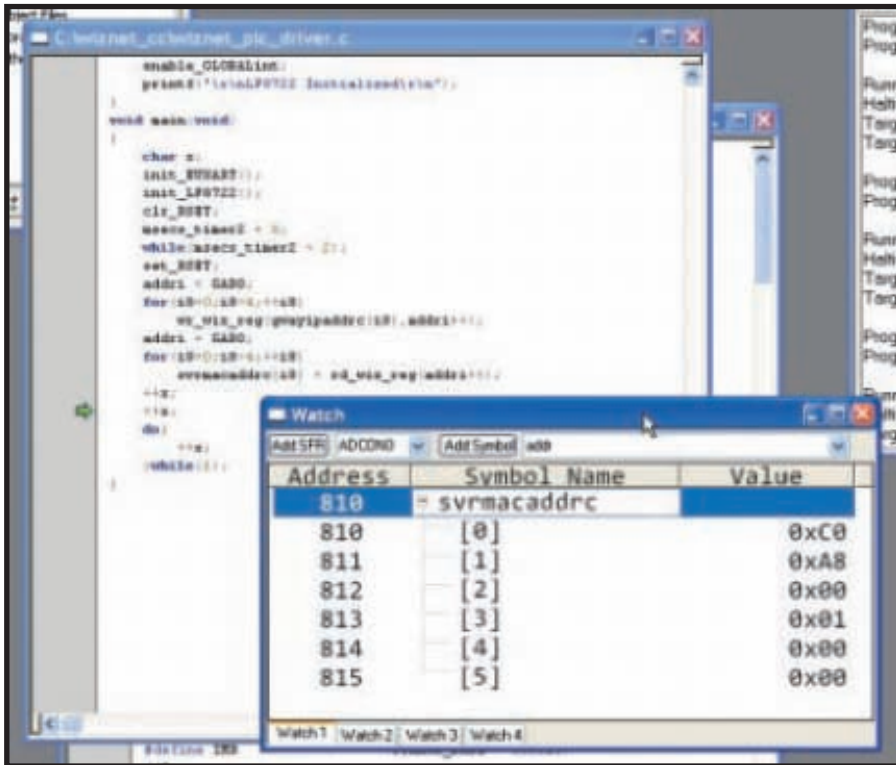


Photo 3—With the success of reading back what I stored in the WIZnet gateway IP address register, we've established a base of operations for reading and writing the W5100's internal registers.

written for AVR devices. So, rather than build my own PIC W5100 include file, I de-Atmeled the factory-supplied W5100 include file. Right now, all I really want from the W5100 include file is the definition code that lays out all of the W5100's internal register addresses. The W5100 factory include file also contains definitions of all of the W5100 register contents, which I'm sure will come in handy later. I lost a bunch of time chasing my soldering snafu, but I gained some of that time back by Microchipizing the original AVR include file.

My first official W5100 firmware act was to punch the W5100 into a hardware reset (see Listing 1). Since I went to all of the trouble to fix those W5100 address and data solder joints, I'm going to run in W5100 Direct Bus Interface mode. Running in Direct Bus Interface mode means that I don't have to touch the W5100 Mode register, which happens to be the very first W5100 Common register. So, we can run our initial W5100/PIC18LF8722 I/O test on the set of Gateway Address registers at address range 0x0001:0x0004. The Gateway Address register addresses GAR0:GAR3 are defined in the include file I converted, which I renamed w5100_pic.h. As

you can see in Listing 1, I put together some basic PIC18LF8722 I/O routines to read and write the W5100 registers. Then, I wrote the contents of the gwayipaddr array to the W5100's Gateway Address Common register set. To make sure I performed the Common register write, I turned around and read the contents of GAR0:GAR3 into an array called svrmacaddr. You can imagine how pleased I was to see the gateway IP address represented in hexadecimal format in the MPLAB WATCH window shot you see in Photo 3. I tested a bit further by using my W5100 register read routines to read the RTR0 Common register pair, which defaults to 0x07D0 and the RCR Common register that follows and defaults to 0x08. All went well. So, I initialized the W5100's gateway, MAC address, subnet mask, and IP address Common registers.

The next step on our way to putting the W5100 project board online involves setting up and defining the socket memory information. We'll use the default of 2-KB-per-socket sizing, which means we don't touch the RMSR (RX memory size) and TMSR (TX memory size) default register values (0x55). As you can see in Listing 2, all we are really

doing is establishing the receive and transmit memory boundaries for each of the four sockets that the W5100 supports. With the socket memory allocation task behind us, we can concentrate on what it takes to manipulate a W5100 socket.

The topmost portion of Listing 3 is the code we will execute to open a W5100 UDP socket. The first order of business is to tell the W5100 what type of socket we want to work with. We are working with UDP at the moment. So, I loaded the socket 0 Mode register with a UDP socket value. I already have an application (EDTP Internet test panel) that will send ASCII characters to well-known port 7 and, as you can see in Listing 3, I've loaded the socket 0 Source Port register with 0x0007. We've already loaded our IP and MAC information. Thus, the addition of the UDP source port value enables us to open a UDP socket. From the proliferation of zeros in the Listing 3 socket initialization code, it should be obvious that we will open the W5100's socket 0 in UDP mode.

Once the socket comes online, we have the power to send and receive UDP datagrams. There are a couple of ways to sense an incoming UDP datagram. We can poll the socket's Received Size register or look for the RECV bit in the socket's Interrupt register. As you can see in the UDP datagram receive code that occupies the center section of Listing 3, I have chosen to use the latter.

An incoming UDP datagram sets the RECV bit of the socket's Interrupt register. Our first reaction to this is to clear the RECV bit by writing a "1" to correspond to the RECV bit's position within the Interrupt register. The W5100 takes care of checksums internally and we, as programmers, never see them in our UDP datagram information. The size of the incoming UDP datagram is automatically posted in the socket's Receive Size register. Here, we retrieve the contents of the Receive Size register and place the value into the get_size variable. I used the W5100's datasheet variable names where possible to make it a bit easier for you to compare my W5100 driver code with the UDP pseudocode flow example in the W5100 datasheet. The receive buffer's read pointer value is kept in the socket's Read Pointer register. We will

Connected. Versatile. Cost-effective.



An embedded Web server out-of-the-box.

Stellaris LM3S6965 Evaluation Kit
featuring integrated Ethernet.

\$69

Stellaris® Means:

- Deterministic ARM® Cortex™ - M3 cores
- Real-time multi-tasking MCUs
- Compatibility from \$1 to 1 GHz



A CAN network out-of-the-box.

Stellaris LM3S2965 Evaluation Kit
featuring integrated CAN.

\$79



Stellaris LM3S8962 Evaluation Kit
featuring integrated Ethernet and CAN.

\$89



Stellaris LM3S811 Evaluation Kit

Discover the exciting ways this evaluation kit was used in DesignStellaris2006 at: www.LuminaryMicro.com/DesignStellaris2006.

\$49



Stellaris LM3S1968 Evaluation Kit
featuring battery backed hibernation.

\$59

Discover what Stellaris can mean for you:
www.LuminaryMicro.com/StellarisMeans



LUMINARYMICRO™



LUMINARY MICRO is the creator of the award-winning Stellaris family of microcontrollers. For more information about the entire Stellaris family of products, please visit our website at LuminaryMicro.com.



use the read pointer value to form the basis for the variable `get_offset`, whose value we will combine with the socket's receive buffer base address to calculate the beginning address of the UDP datagram's header. The UDP datagram header offered by the W5100 is made up of 4 bytes of destination IP address, 2 bytes of destination port address, and 2 bytes of data size information. Thus, the `header_size` variable value is eight. Once all of the addressing calculations have been made, we can use our W5100 read register routine to store the data away in the PIC18LF8722's SRAM for later.

Logically, what is not header information must be data information because we are protected from checksums by the W5100 architecture. With that, we can deduce that the `udp_data_size` variable will contain the number of data bytes we need to retrieve and store. Again, using our home-brewed W5100 I/O code, we read the data from the W5100 receive buffer memory and store it in the appropriate PIC18LF8722 SRAM locations. Our

Listing 2—The W5100 datasheet talks about this with pseudocode. Here's my translation.

```
#define chip_base_address      0x0000
#define RX_memory_base_address 0x6000
#define gS0_RX_BASE          chip_base_address + RX_memory_base_address
#define gS0_RX_MASK          0x0800 - 1
#define gS1_RX_BASE          gS0_RX_BASE + (gS0_RX_MASK + 1)
#define gS1_RX_MASK          0x0800 - 1
#define gS2_RX_BASE          gS1_RX_BASE + (gS1_RX_MASK + 1)
#define gS2_RX_MASK          0x0800 - 1
#define gS3_RX_BASE          gS2_RX_BASE + (gS2_RX_MASK + 1)
#define gS3_RX_MASK          0x0800 - 1
#define TX_memory_base_address 0x4000
#define gS0_TX_BASE          chip_base_address + RX_memory_base_address
#define gS0_TX_MASK          0x0800 - 1
#define gS1_TX_BASE          gS0_TX_BASE + (gS0_TX_MASK + 1)
#define gS1_TX_MASK          0x0800 - 1
#define gS2_TX_BASE          gS1_TX_BASE + (gS1_TX_MASK + 1)
#define gS2_TX_MASK          0x0800 - 1
#define gS3_TX_BASE          gS2_TX_BASE + (gS2_TX_MASK + 1)
#define gS3_TX_MASK          0x0800 - 1
```

absorption of the UDP datagram and its header is complete. We end our receive session by issuing the `RCV` command in the socket's Command register, which updates the receive buffer pointers. For those of you who are following along with the pseudocode flow in the W5100 datasheet, note that the `udp_data_size` variable is not a W5100 datasheet

variable. It's a Fred variable.

Sending a UDP datagram is very similar to receiving one. We'll reuse the information we received earlier and bounce a UDP message back at the sender. Recall that our received UDP datagram header contained a destination IP address and a destination UDP port value. We thought ahead and stored both of the header values. Now all we have to do is retrieve them from the PIC18LF8722's SRAM and load them into the proper W5100 Socket registers. I begin my UDP datagram transmission in that manner within the bottom portion of the code in Listing 3. Using the socket's transmit buffer write pointer, I calculate where in the W5100 transmit buffer to begin stuffing the data I wish to transmit. I then transfer the previously stored UDP datagram data from the PIC18LF8722's SRAM into the W5100's transmit buffer. The W5100 will transmit the data located between the socket's transmit read pointer and transmit write pointer. So, I must update the transmit write pointer by increasing it by the number of bytes I need to transmit. Once that's done, I issue the `SEND` command and wait for the send success signal, which is a cleared socket Command register. I can now issue a `CLOSE` command in the socket's Command register to close the socket or send or receive another UDP datagram.

CONGRATULATIONS!

You have completed W5100 bootcamp. In addition to the W5100 register I/O

Intelligent Display Engine
IntelliLCD

- 7" Wide TFT Color Touch Screen
- Supports TrueType Windows Fonts
- RS232 Interface
- 800 x 480 resolutions
- 260k Colors
- SD Card Support
- BMP File Support
- LED Backlight
- USB PC sync, keyboard, mouse
- Display Commands: Line, Circle, Box, Print...
- Control Commands: Button, Slider, CheckBox...

\$399
cty 1

COMFILE TECHNOLOGY www.comfiletech.com
Toll-Free: 1.888.928.2562

Educators and Students: Register Today

to Receive Exciting
Microcontroller Resources!

RENEASAS
UNIVERSITY
PROGRAM



Renesas — the #1 supplier of microcontrollers in the world — is launching Renesas University, an exciting educational program that gives educators a way to teach microcontroller (MCU) technology using a modern architecture and professional-grade tools. It also offers many valuable resources that help students learn about MCUs and how they can be applied in significant embedded system designs.

#1 MCU
REACH
FURTHER

Renesas is a worldwide leader in:

- ▶ Microcontrollers
- ▶ Embedded flash microcontrollers
- ▶ MCUs in car navigation systems
- ▶ Power amplifiers for GSM phones
- ▶ LCD controllers for color mobile displays

The Renesas University program nurtures an online community where educators and students come together to share ideas, address technical issues and discuss microcontroller topics. It is characterized by:

Publish ▶ Renesas actively encourages academics and students to publish microcontroller-related papers. We provide assistance in publishing course material and microcontroller related books.

Toolchain ▶ The Renesas integrated development environment with toolchain is the commercial version of our development tools – with full C compiler, assembler, linker, and debugger. It is not a typical capability-reduced “educational” version. The only limitation is a 64KB code size after 60 days of use.

Modern ▶ Renesas microcontrollers utilize a modern architecture designed specifically for C and other high-level languages. Our devices handle the most demanding applications of today and tomorrow.

Complete Development Kits



Renesas Starter Kits provide a USB-powered, MCU-based system board with in-circuit debugger/flash memory programmer. They include a CD containing our integrated development environment with toolchain, plus documentation, example firmware and interesting projects.

- ▶ **Free for Educators:** Register at the Renesas University website to receive ten free Starter Kits per semester. In return, we request the submission of material that enriches Renesas University; i.e., code, student projects, technical papers, embedded control designs, etc.
- ▶ **Low cost for Students:** If actively enrolled in an educational institution, a Starter Kit can be purchased at a very low cost after registering at the Renesas University website.

For more information on Renesas University and how to enroll, please visit www.renesasuniversity.com or email: University@rta.renesas.com



© 2007 Renesas Technology America, Inc. Renesas Technology America, Inc. is a wholly owned subsidiary of Renesas Technology Corp.

Everywhere you imagine. **RENEASAS**

Only 4 Steps...

...are required to generate efficient, reliable applications with the μ Vision 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 μ Vision, select the microcontroller to pre-configure tools and obtain CPU startup code.

Step 2. Configure the Device and Create Application Code

The μ Vision 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.

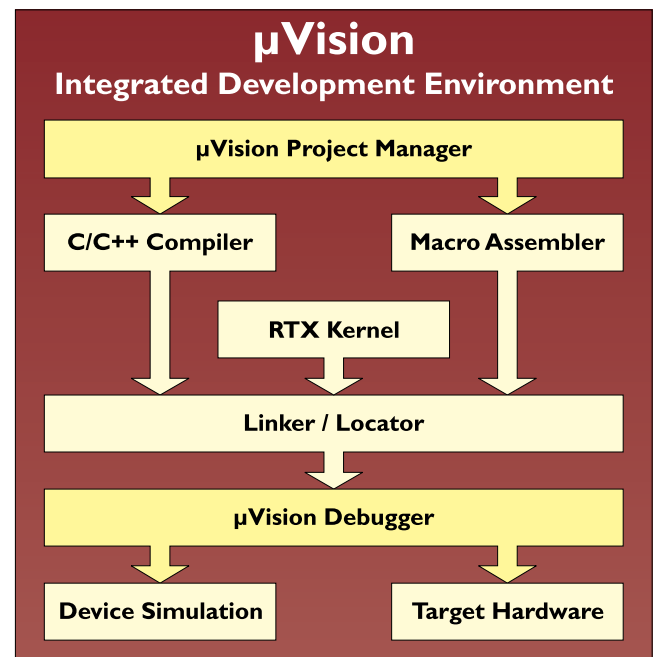


Step 4. Download to Flash and Test Application

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


Listing 3—Think about it. All you ever do with any communications device is receive and transmit. I pulled the logic behind this code from the pseudocode flow in the W5100's datasheet.

```
//SOCKET INTI*****
do{
wr_wiz_reg(Sn_MR_UDP,Sn_MR(0)); //protoco1 = UDP
wr_wiz_reg(0x00,Sn_PORT0(0)); //well-known ECHO port
wr_wiz_reg(0x07,Sn_PORT1(0));
wr_wiz_reg(Sn_CR_OPEN,Sn_CR(0)); //give the open command
if(rd_wiz_reg(Sn_SR(0)) != SOCK_UDP) //wait for the socket to come online
wr_wiz_reg(Sn_CR_CLOSE,Sn_CR(0));
}while(rd_wiz_reg(Sn_SR(0)) != SOCK_UDP);

//RECEIVE*****
do{
//look for incoming UDP datagrams
i16 = rd_wiz_reg(Sn_IR(0));
}while(i16 == 0);
wr_wiz_reg(0x04,Sn_IR(0));
//get the datagram size
hi_byte = rd_wiz_reg(Sn_RX_RSR0(0));
lo_byte = rd_wiz_reg(Sn_RX_RSR1(0));
get_size = make16(hi_byte,lo_byte);
//get the datagram's buffer offset
hi_byte = rd_wiz_reg(Sn_RX_RD0(0));
lo_byte = rd_wiz_reg(Sn_RX_RD1(0));
get_offset = make16(hi_byte,lo_byte) & gS0_RX_MASK;
//calculate the datagram's starting buffer address
get_start_address = gS0_RX_BASE + get_offset;
//UDP header size
header_size = 8;
//store the UDP header information
addri = get_start_address;
for(i8=0;i8<header_size;++i8)
{
packet[ip_destaddr+i8] = rd_wiz_reg(addri++);
++get_offset;
}
//store the UDP data
get_start_address = gS0_RX_BASE + get_offset;
udp_data_size = get_size - header_size;
addri = get_start_address;
for(i8=0;i8<udp_data_size;++i8)
{
packet[UDP_data+i8] = rd_wiz_reg(addri++);
++get_offset;
}
//update the receive buffer pointers
wr_wiz_reg(Sn_CR_RECV,Sn_CR(0));

//TRANSMIT*****
//load destination IP address
addri = Sn_DIPRO(0);
for(i8=0;i8<4;++i8)
wr_wiz_reg(packet[ip_destaddr+i8],addri++);
//load destination port address
addri = Sn_DPORT0(0);
for(i8=0;i8<2;++i8)
wr_wiz_reg(packet[UDP_srcport+i8],addri++);
//get transmit buffer offset
hi_byte = rd_wiz_reg(Sn_TX_WR0(0));
lo_byte = rd_wiz_reg(Sn_TX_WR1(0));
get_offset = make16(hi_byte,lo_byte) & gS0_TX_MASK;
//calculate transmit data buffer start address
get_start_address = gS0_TX_BASE + get_offset;
//load data into transmit buffer
addri = get_start_address;
for(i8=0;i8<udp_data_size;++i8)
{
wr_wiz_reg(packet[UDP_data+i8],addri++);
++get_offset;
}
//update transmit buffer pointer
wr_wiz_reg((make8(get_offset,1)),Sn_TX_WR0(0));
wr_wiz_reg((make8(get_offset,0)),Sn_TX_WR1(0));
//send data
wr_wiz_reg(Sn_CR_SEND,Sn_CR(0));
while(rd_wiz_reg(Sn_CR(0)));
```

code, UDP transmit code, and UDP receive code, you have a basic and flexible W5100 hardware design to work with.

Here's a hint that will help you determine very early on where your W5100 design stands: Execute only the code through loading the IP address. Load the gateway address, the MAC address, the subnet mask, and the IP address. Don't open any sockets. At this point, you can PING your W5100 design. If you get good PING returns, your PHY hardware and your W5100 register I/O code are good to go. You've also tested and confirmed the operation of your W5100 address, data, and control signals. Bringing up UDP is a fun and easy way to get to know the W5100. The EDTP Internet test panel is a UDP application that runs on your PC. The EDTP Internet test panel is available for download from www.edtp.com. 

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/2007/208.

SOURCES

EDTP Internet test panel

EDTP Electronics, Inc.
www.edtp.com

HI-TECH PICC-18 C Compiler

HI-TECH Software
www.htsoft.com

PIC18LF8722 Microcontroller, REAL ICE, and MPLAB

Microchip Technology, Inc.
www.microchip.com

RDI-125BAG1A Pulse transformer

U.D. Electronic Corp.
www.ude-corp.com

W5100 TCP/IP Ethernet controller

WIZnet, Inc.
www.ewiznet.com

Hybrid Computing on an FPGA

Bruce explains how to simulate the parallel functions of an analog computer on an FPGA. Now you can harness the advantages of parallel execution and a general-purpose CPU on the same chip.

When I was a freshman taking physics in 1964, I had to analyze coupled, spring-mass, harmonic oscillator data. There was one digital computer available (for the entire school!) and one analog computer. The line was long for the card punch, and FORTRAN ran slowly on a machine that did about 100 multiplies/second, so I decided to learn to use the analog computer. The analog computer was faster at the time for solving differential equations because it was fully parallel. Each separate addition, integral, and multiply happened at the same time, so results were immediately available on the attached analog pen plotter.

Now I teach a course at Cornell University (ECE 576) in which I use field-programmable gate arrays (FPGAs) to build embedded systems, which may include processor cores, DSP units, I/O units, and other specialized processing. To get a good performance from an FPGA, you need to execute as many operations as possible in parallel because the clock rate is relatively slow compared to a dedicated CPU. I wondered if I could simulate the parallel functions of an analog

$$\begin{aligned} v1(n+1) &= v1(n) + dt \times (f1(t, v1(n), v2(n), v3(n), \dots, vm(n))) \\ v2(n+1) &= v2(n) + dt \times (f2(t, v1(n), v2(n), v3(n), \dots, vm(n))) \\ v3(n+1) &= v3(n) + dt \times (f3(t, v1(n), v2(n), v3(n), \dots, vm(n))) \\ &\dots \\ vm(n+1) &= vm(n) + dt \times (fm(\dots)) \end{aligned}$$

Figure 2—The equations in Figure 1 are solved digitally by taking small, discrete time steps while updating all of the state variables at each step.

$$\begin{aligned} \frac{dv1}{dt} &= f1(t, v1, v2, v3, \dots, vm) \\ \frac{dv2}{dt} &= f2(t, v1, v2, v3, \dots, vm) \\ \frac{dv3}{dt} &= f3(t, v1, v2, v3, \dots, vm) \\ &\dots \\ \frac{dvm}{dt} &= fm(\dots) \end{aligned}$$

Figure 1—These are differential equations to be solved by hybrid computation. These equations could result from a physical simulation or game environment. The $v1$ to vm are referred to as state variables and could represent position, velocity, or voltage.

computer on an FPGA and get the advantages of parallel execution while being able to use the extremely handy ability to implement a general-purpose CPU on the same chip to control the “analog” simulation.

HARDWARE & SOFTWARE

Several courses at Cornell use the Altera DE2 educational board. The board has a fairly large Altera Cyclone II FPGA and a wide range of hardware interfaces, including LEDs, switches, an audio I/O codec, an NTSC video input codec, a VGA output, USB interfaces, an SD card interface, an Ethernet controller, a serial port, an IrDA interface, 512 KB of SRAM, 8 MB of SDRAM, 4 MB of flash memory, and 80 lines of general-purpose lines arranged as two 40-pin connectors. The board costs \$269 for academic users.

The board's Cyclone II FPGA has about 33,000 logic elements,

each of which can be configured as an arbitrary logic function four-input gate with an output register bit and support for arithmetic. Logic elements are connected via switches to each other, to memory, and I/O to form a specified circuit. Implementing a 32-bit CPU takes less than 10% of the logic units. In addition to the general-purpose logic units, there are 105 blocks of 4-Kb memory, 35 18-bit hardware multipliers, and four phase-locked loops.

To control all of this hardware, there is a downloadable suite of software called Quartus II for Education and Nios II for Education. Quartus II enables a student to design hardware in Verilog, VHDL, or by schematic, and includes point-and-click CPU design for the Nios II soft processor. The Nios II software provides a GCC development environment for the processor you just built, including a custom C library specific to your design. In 2004, George Martin wrote two articles explaining how to build a Nios processor in the FPGA and how to

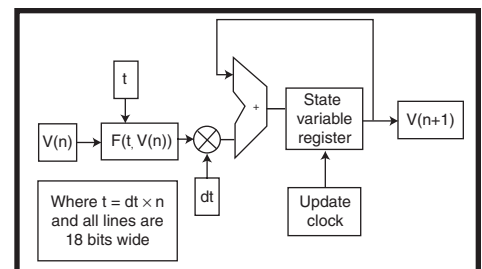


Figure 3—This is a block diagram of the Euler integrator. At each time step, an input function is multiplied by dt and added to the sum of past output values to perform a numerical integration over time.

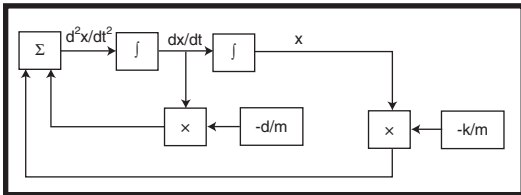


Figure 4—This is a block diagram showing how integrators (like those defined in Figure 3) and multipliers could be connected to solve a second-order differential equation. Inputs for the time step $dt(t)$ are not shown but would be connected to each integrator.

program the processor in GCC (“Designing with the Nios,” Parts 1 & 2, *Circuit Cellar* 167 & 168, 2004).

I have found that the Verilog hardware definition language is good for advanced design because it is less bulky than VHDL, but it’s easier for big designs than schematic capture. A module of Quartus II, called SPOC builder, generates a CPU design in Verilog based on a series of specifications, which the student chooses. The specifications can include the number and bit width of I/O ports, the number of hardware timers, memory type (on chip, SRAM, SDRAM), and size to be used (i.e., cache size and pipeline options). Multiprocessor designs are supported with a defined bus structure, shared hardware mutex structures, and shared mailboxes. It is possible to design and build a fully functional 32-bit CPU in a few minutes.

SIMULATE ANALOG COMPUTATION

Traditionally, analog computers have used op-amps to implement addition and subtraction and to compute time integrals of functions. Multiplication was handled by analog multipliers. All of the operations are easy to perform on the FPGA, except for computing time integrals. However, around the time that digital circuitry was getting faster than analog computation, a device called a digital differential analyzer (DDA) was invented to simulate the integral function digitally. I used a version of the DDA, which was easy to implement on the FPGA and gave speedy execution.

But first, what problem are

you trying to solve? Typically, physical simulations involve the solution of differential equations. The differential equations fall directly out of Newton’s $F = ma$ because forces are often related to position (e.g., gravity or spring) or velocity (e.g., frictional drag) and acceleration is the second derivative of position. If you

define a set of state variables to be the position and velocity of each particle of interest, then the system behavior will be determined by a set of differential equations with state variables v_1 to v_m , and with arbitrary functions relating combinations of all the state variables to the rate of change of each state variable (see Figure 1).

To solve the equations in Figure 1 digitally, you need to take discrete time steps. If the time step is of duration dt and you index time by n , then you can use the values of the state variables at time n to compute the values at time $n + 1$ and then repeat. You will build circuitry to perform an Euler integration approximation to the equations in Figure 2.

The number system you will use is 18-bit 2’s complement and fixed point. It is also compatible with the hardware multipliers and standard addition and subtraction. Numbers are scaled so there are 16 bits of fraction, with a sign bit and one bit of integer with a range of 1.999985 to -2.0000 . Real

analog computers also require amplitude scaling because they are bound at high voltage by physical limits and at low voltage by noise.

Figure 3 shows the structure of the numerical integrator as a block diagram. Many copies of the circuit would be built on the FPGA for a real application and all would perform their operations at the same time. Quite often, calculating $F(t, V(n))$ is more complicated than the integration itself.

As a simple example, consider the linear, second-order differential equation resulting from a damped spring-mass system:

$$\frac{d^2x}{dt^2} = \frac{-k}{m} \times x - \frac{d}{m} \times \left(\frac{dx}{dt}\right) \quad [1]$$

where k is the spring constant, d is the damping coefficient, m is the mass, and x is the displacement of the mass. You will simulate this by converting the second-order system into two first-order equations. If you let $v_1 = x$ and $v_2 = dx/dt$, then the second-order equation is equivalent to:

$$\begin{aligned} v_2 &= \frac{dv_1}{dt} \\ \frac{dv_2}{dt} &= \frac{-k}{m} \times v_1 - \frac{d}{m} \times v_2 \end{aligned} \quad [2]$$

The equations can be solved by wiring two integrators, two multipliers, and an adder together (see Figure 4).

Controlling such a system to start and stop time, record variable values, and load initial conditions is easily done using a standard sequential computer. The next sections will outline the actual hardware built using Verilog to specify multiple integrators, adders, multipliers, and one CPU to control it all.

Verilog is a text-based hardware description language. Superficially, it looks like a sequential programming language, but it is not. All statements in a Verilog design execute at the same time! Think of each statement as defining the values on a wire or bus. Modules,

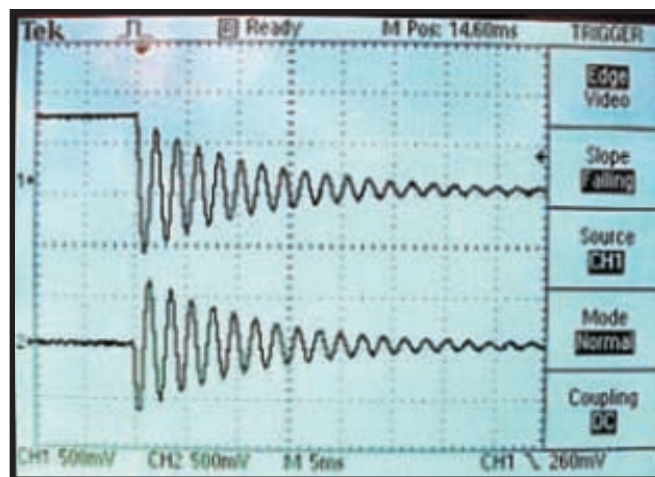


Photo 1—Here is the impulse response of the simulated second-order system. The spring-mass system was released from a fixed position at $t = 0$ and allowed to bounce. The top trace is position and the bottom is velocity. Output to the scope is through the DE2 audio codec.

Use	Module Name	Description	Input Clock	Base	End	IRQ
<input checked="" type="checkbox"/>	cpu_0	Nios II Processor - Altera Corporation	clk			
	instruction_master	Master port				
	data_master	Master port				
	jtag_debug_module	Slave port				
<input checked="" type="checkbox"/>	jtag_uart_0	JTAG UART	clk	0x00000000	0x000007FF	IRQ 0
<input checked="" type="checkbox"/>	timer_0	Interval timer	clk	0x00000800	0x00000807	IRQ 31
<input checked="" type="checkbox"/>	DDS_incr	PIO (Parallel I/O)	clk	0x00000820	0x0000083F	
<input checked="" type="checkbox"/>	control	PIO (Parallel I/O)	clk	0x00000810	0x0000081F	
<input checked="" type="checkbox"/>	input_gain	PIO (Parallel I/O)	clk	0x00000840	0x0000084F	
<input checked="" type="checkbox"/>	phase	PIO (Parallel I/O)	clk	0x00000850	0x0000085F	
<input checked="" type="checkbox"/>	amplitude	PIO (Parallel I/O)	clk	0x00000860	0x0000086F	
<input checked="" type="checkbox"/>	sdram_0	SDRAM Controller	clk	0x00000870	0x0000087F	
			clk	0x00000000	0x00FFFFFF	

Photo 2—This is the user interface of the Quartus II SOPC builder. The user drags microcontroller modules, a memory interface, a JTAG interface, and I/O ports onto this list and “wires” them together by clicking on the bus icons to the left side. The names chosen in the interface are picked up by the C compiler when it generates I/O macros for programming.

which superficially resemble subroutines, cause hardware to be built and connected every time their name is used in a design.

Starting at the bottom of the DDA design, consider the multiplier. You are using 18-bit fixed point with the binary point between bits 15 and 16 for 16 bits of fraction, one bit of integer, and a sign bit. The two 18-bit numbers result in a 36-bit result, of which bit 35 is the new sign bit and the rest of the value is in bits 32 to 16. The Verilog is shown in Listing 1. A module designed in this fashion in Quartus II will result in using one of the hardware multipliers in the FPGA, according to the Altera HDL guidelines manual. Each time the module is invoked, a new multiplier will be built.

The integrator module updates a state variable register (v1) on the positive edge of the system clock. The multiply of the input function by dt is simplified to a shift-right. This works because the value of dt is less than one and because steps of two are fine enough control. The strange triple-arrow operator for shift means “shift-right-signed” (see Listing 2).

The second-order example in Figure 4 can now be coded as shown in Listing 3. Three signed multipliers are built and two integrators. The form 18’h0_0800 means an 18-bit hexadecimal constant with value 1/32 in the notation used (see Listing 3).

The outputs of both the integrators (velocity and position) were wired to the audio codec (a Wolfson Microelectronics WM8731) stereo outputs and displayed on a scope. The 18-bit 2’s complement notation used in the calculation was truncated to 16 bits when

sent to the codec. Refer to Photo 1 for the output. The top trace is position and the bottom is velocity. The characteristic damped sine wave of a second-order system can also be seen. With a $dt = 2^{-9}$, the system runs 32 times faster than real time and had to be slowed down to use the audio codec. The resonant frequency of the simulated oscillator matched the analytical value within 0.2%.

It is often useful to control an ana-

log computer with a digital computer. Such a system is called a hybrid computer. For example, you might want to sequence through a set of input frequencies applied to the simulated system to build a Bode plot. A Nios II CPU was built on the FPGA with ports to control the DDS frequency, control the gain of the sine wave applied to the second-order system, control the analog reset and start the simulation, and record the amplitude and phase shift of the system under test. The top-level Verilog module contains the DDA simulation of the second-order system and the Nios II CPU. The description of the Nios II CPU is shown in Photo 2, which is a screen dump of the Quartus II SOPC builder for the CPU used in this design. Designing the CPU consisted of drag-and-drop operations on the table. The module name column shows the name I assigned to each I/O port. The names will be picked up by

Listing 1—This Verilog module implements a signed, 18-bit, fixed-point multiply with 16 bits to the right of the binary point. When implemented with this syntax, the Quartus II software instantiates the circuit using hardware multipliers.

```

module signed_mult (out, a, b);
    output [17:0] out;
    input signed [17:0] a;
    input signed [17:0] b;
    wire signed [17:0] out;
    wire signed [35:0] mult_out;
    assign mult_out = a * b;
    assign out = {mult_out[35], mult_out[32:16]};
endmodule

```

Listing 2—This Verilog module implements a signed 18-bit Euler integrator. Input parameters include a fractional dt (delta t) value represented as the number of shifts right to be applied. The reset input causes the module to output the InitialOut value.

```

module integrator(out,funct,InitialOut,dt,clk,reset);
    output [17:0] out; //the state variable V
    input signed [17:0] funct; //the dV/dt function
    input [3:0] dt; // in units of SHIFT-right
    input clk, reset;
    input signed [17:0] InitialOut; //the initial state variable V
    wire signed [17:0] out, vnew;
    reg signed [17:0] v1;
    always @ (posedge clk)
    begin
        if (reset==0) //reset
            v1 <= InitialOut; //
        else
            v1 <= vnew;
    end
    assign vnew = v1 + (funct>>>dt);
    assign out = v1;
endmodule

```


What a Simple

Born to Internet!

'3-in-1' chip solution

= Hardwired TCP/IP + MAC + PHY



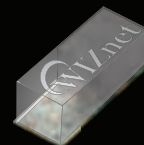
Serial-to-Ethernet
WIZ100SR



Serial-to-Ethernet
WIZ110SR



All-in-One Module
WIZ810MJ



All-in-One Connector
WIZ800MJ
Coming Soon

How to buy : <http://www.wiznet.co.kr/how.htm>



Put your Creativity to the Test!

Hardwired TCP/IP solution company,
WIZnet holds iEthernet **Design Contest**
with CIRCUIT CELLAR

- **Item:** W5100 or WIZ modules with W5100
- **Period:** 2007.9.20 ~ 2008.1.31
- **Official web site:** www.circuitcellar.com/wiznet
- **Winners Announce:** Circuit Cellar April, 2008
- **Display:** ESC SV 2008 (4.14~4.18) at WIZnet Booth

**CIRCUIT
CELLAR**
THE MAGAZINE FOR COMPUTER APPLICATIONS

WIZnet 2007
iEthernet
DESIGN CONTEST

* Your project could win you a share of \$15,000 in cash prizes and give you the international recognition you deserve.

WIZnet

www.wiznet.co.kr
www.ewiznet.com

Listing 3—The Verilog code fragment instantiates two integrators and three multipliers to implement the damped spring-mass simulator. The initial conditions are zero and the dt is 2^{-9} . The second-order system is driven by a sine wave generated by a direct digital synthesis unit elsewhere in the hardware. (The full code can be downloaded from the *Circuit Cellar* FTP site).

```
// wire the integrators
// time step: dt = 2<<9
// v1(n+1) = v1(n) + dt*v2(n)
integrator int1(v1,v2,0,9,AnalogClock,AnalogReset);
// v2(n+1) = v2(n) + dt*(-k/m*v1(n) - d/m*v2(n))
signed_mult K_M(v1xK_M, v1, 18'h1_0000); //Mult by k/m
signed_mult D_M(v2xD_M, v2, 18'h0_0800); //Mult by d/m
//scale the input so that it does not saturate at resonance
signed_mult Sine_gain(Sinput,{sine_out[15],sine_out[15],sine_out},
{2'h0,Sgain});
integrator int2(v2,(-v1xK_M-v2xD_M+Sinput),
0,9,AnalogClock,AnalogReset);
```

Listing 4—This module is the Verilog description of the Nios II CPU interface. The clock, reset line, I/O ports, and SDRAM interface are defined here. The module is generated by the Quartus II SOPC builder based on a high-level (point-and-click) user specification.

```
module hybrid_cnt1 (
// 1) global signals:
clk,reset_n, out_port_from_the_DDS_incr, in_port_to_the_amplitude,
out_port_from_the_control, out_port_from_the_input_gain, in_port_to_the_phase,
// the_s dram_0
zs_addr_from_the_s dram_0,
zs_ba_from_the_s dram_0,
zs_cas_n_from_the_s dram_0,
zs_cke_from_the_s dram_0,
zs_cs_n_from_the_s dram_0,
zs_dq_to_and_from_the_s dram_0,
zs_dqm_from_the_s dram_0,
zs_ras_n_from_the_s dram_0,
zs_we_n_from_the_s dram_0
)
```

the Nios II GCC tools and used to build a C library for the configuration. This column also shows how each module is connected to data and instruction buses. The “base” and “end” columns show the address map for each module. The IRQ column shows any interrupt vectors associated with each module.

The CPU module generated by the SOPC builder is completely defined by a Verilog description, which is instantiated in the same module that defines the DDA hardware. There are nine control ports for the SDRAM, as well as the clock, reset, and the I/O ports I defined (see Listing 4).

The GCC program running on the Nios II: holds the DDA in reset and initializes the test frequency, releases the DDA reset, waits until the DDA output phase and magnitude reach steady-state, prints the phase and magnitude through the JTAG UART, increments the frequency by a small factor, and repeats the above steps for a number of frequencies.

A set of C macros generated by the Nios II development environment provides interfaces to the hardware. For instance, to write a zero to the I/O port named “control” in the SOPC builder, you could use the following statement:

```
IOWR_ALTERA_AVALON_PIO_DATA(CONTROL_BASE,0);
```

The Avalon notation refers to the bus structure used by the Nios II CPU. The CONTROL_BASE is the first address of the address map of the “control” port.

The resulting phase/magnitude Bode plots in Photo 3 were produced with a small Matlab script that read the text file generated by the Nios II and superimposed the analytical solution generated by Matlab on a PC. The top plot is log amplitude versus log frequency and the bottom is phase versus log frequency. You can see that the match to the analytical solution (red curves) is good. The resonant frequency can be easily seen and the analytical value is marked with a vertical red line. You can also see some amplitude quantization

How would you use a Rejutor™?

Enter the Design Challenge and Win!

a 2008 Mini Cooper, 2 Harleys, cash and more

(see www.rejutorchallenge.com for details)



Tell us about your design ideas for a Rejutor and be entered to win a Mini Cooper, or not one but two Harleys or cash as first prize. Also multiple second and third prizes will be awarded.

As the name suggests, Rejutors are highly precise re-adjustable resistors. Visit www.microbridgetech.com for product specifications.

888-Rejutor (735-8786)



The Rejutor Company

Harley-Davidson Motor Company, Nintendo, Circuit Cellar, Inc. and BMW are not endorsing, sponsoring, or otherwise affiliated with this promotion. © 2007 Microbridge Technologies Corp. Rejutor is a trademark of Microbridge Technologies Corp.

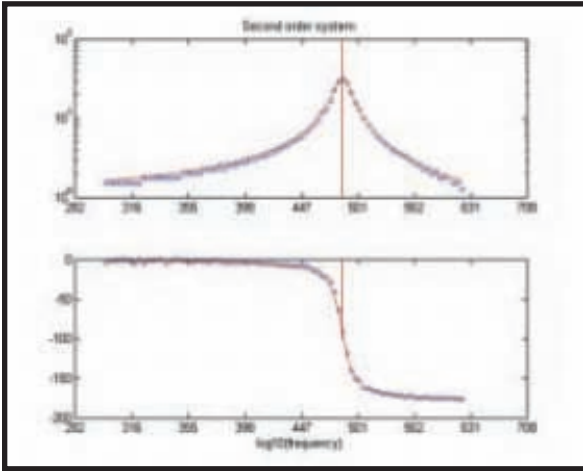



Photo 3—These are the magnitude and phase Bode plots for a simulated second-order system. The circles are data collected by the Nios II CPU from the simulated analog system. The red line is the exact solution computed in Matlab on the PC.

error at low amplitudes on both ends of the curve.

ANALOG VS. FPGA

Each integrator takes about 2% of the circuitry on the FPGA and each multiplication takes 3% of the multipliers. Thus, you could expect to build about 50 integrators and about 30 multipliers in a larger design. If you filled the FPGA with integrators and multipliers and ran them at 50 MHz, you could expect to compute around four billion 18-bit operations per second. Of course, putting a control CPU on the FPGA would drop the number of integrators you could build.

Let's compare the experience of programming an analog computer by plugging in resistors and wires in 1964 and using Verilog and Quartus II to build one on a Cyclone II FPGA in 2007. The learning and set up time for my first design on both systems was about the same (a few days). The set up time for the second project on the FPGA was much shorter. Adding integrators in a text file is faster than running wires. The FPGA version has a simulation bandwidth at least 500 times higher, computed as (number of integrators) \times (integrator bandwidth). The analog computer ran in "continuous time" and thus had no time-quantization error but had no better than 1% (7-bit) amplitude accuracy. The FPGA simulation seems to have an accuracy of 18 bits, but it is actually somewhat lower, depending on the size of dt.

Picking a small dt decreases the time-quantization error, but increases the cumulative round-off error. Extending the numerical precision or using a higher-order integrator would help soften this trade-off. 

Bruce Land (brl4@cornell.edu) is a senior research associate in both Neurobiology and Behavior and Electrical and Computer Engineering at Cornell University. He teaches two courses in Neurobiology and Behavior and

two in Electrical and Computer Engineering. Bruce also provides general research support in electronics design and computer techniques.

PROJECT FILES

To download code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/208.

RESOURCES

Altera Corp., "Recommended HDL Coding Styles," QII51007, 2007, www.altera.com/literature/hb/qts/qts_qii51007.pdf.

B. Land, "DDA on FPGA: A Modern Analog Computer," Cornell University, <http://instruct1.cit.cornell.edu/courses/ece576/DDA/index.htm>.

B. Land, ECE 576 Advanced Microcontroller Systems on a Programmable Chip, Cornell University, <http://instruct1.cit.cornell.edu/courses/ece576>.

G. Martin, "Designing with the Nios," Parts 1 & 2, *Circuit Cellar* 167 & 168, 2004.

SOURCES

DE2 Board, Nios II embedded processors, and Quartus II software
Altera Corp.
www.altera.com

WM8731 Codec
Wolfson Microelectronics, Inc.
www.wolfsonmicro.com

We Listen. Think. And Create.

Distributed I/O

Digital I/O

Serial I/O

Industrial Computing

HMI

SeaLINK USB serial adapters are the fastest, most reliable way to connect peripherals to any USB-equipped computer.



SeaLINK USB Serial Adapters Provide:

- 1, 2, 4, 8, and 16-Port Models
- RS-232, RS-422, and RS-485 Serial Interfaces
- Data Rates to 921.6K bps
- State Machine Architecture to Reduce Host Processor Overhead
- Operation as Standard COM Ports to the Host Computer
- Lifetime Warranty

FOCUS
On Success
Call Today!

SEALEVEL
scalevel.com > sales@scalevel.com > 864.843.8067

Communication Protocols

The next time you choose an error-detection or correction method, consider your application and data needs. Massimo surveys the communications landscape and comments on new protocols, network architecture, and communication software.

For years, UART and USART were the most common communication interfaces because of their simplicity (they didn't require too many transistors on silicon) and applicability (they were used in printers, modems, and terminals). Today, complex communication interfaces, such as Ethernet and USB, are rapidly spreading throughout the embedded market. As you know, it is quite common to find microcontrollers equipped with USB and Ethernet interfaces in addition to UART, USART, I²C, SPI, 1-Wire, and CAN technologies. As a result, you have a wide range of communication interfaces to use in your applications. This requires more complex firmware and technical know-how.

APPLICATIONS & PROTOCOLS

The increasing demand for communication has led to various new communication protocols (e.g., DMX512, which is a data protocol over an RS-485 bus line used to control lights and theatrical devices). There are various reasons to design a new communication protocol. Marketing policies and application requirements are just a few. Some applications require high levels of security, others require speed and reliability. An automatic teller machine (ATM), for example, has to be simple, secure, and fast. It also must be able to manage concurrent transactions. The communication protocol must be just as flexible, especially if the ATM connects to the ATM transaction processor via a dial-up modem over a telephone line.

When dealing with new technologies or troubleshooting older ones, you must sometimes design atypical solutions.

About 10 years ago, I was called to work with a team of engineers on a large video system. It worked well when directly coupled with PAL/NTSC analog video signals, but it couldn't display decent images if they came from a digital source. The communication protocol implemented between the A/D video mixer and the video wall driver was a SEND-ACK half-duplex protocol. Its retransmission timeout was on a dedicated high-speed serial interface (custom-made chips by Philips) on a differential pair cable. The hardware was good (probably one of the best solutions I had ever seen), but the communication protocol was inadequate. Error detection and management weren't as important as a fast, continuous stream of data. I had to design a solution.

I decided to send a complete image at the maximum possible speed without buffering. I used a simplex protocol scheme. It wasn't perfect at first, so I implemented data error encoding and a correction scheme using Golay code, which is a perfect linear error-correcting code that encodes 12 bits of data in a 24-bit word in such a way that any triple-bit error can be corrected and any quadruple-bit error can be detected. For Voyager 1 and 2, NASA used the same approach to transmit images back to Earth. It worked remarkably well.

NEW OR OLD?

Your first dilemma: do you choose a standard protocol or design a new one?

Choosing a standard pro-

col can make your life easier, especially if you go with a complex protocol such as TCP/IP, USB, CANopen, or DeviceNet. There are a lot of good libraries and open-source code at your disposal. But not all standard protocols define an application data protocol (OSI/ISO application layer), so you might have to write some source code. This is especially true if you use the same data protocol over different underlying protocols. Let's consider an example.

What happens when you use an application data protocol over an RS-232 line and a TCP over an Ethernet network? At the application-level layer, sender and receiver nodes have a common API to manage the application data. With respect to the network, the payload data has to travel encapsulated on a TCP/IP packet (over Ethernet) and on an RS-232 packet in the same way and with the same format. Of course, if the physical level of Ethernet and RS-232 are not the same, then the network must have a bridge to translate Ethernet and RS-232.

The TCP/IP protocol stack solutions are SLIP and PPP data link layer

Layers model	Thin-level layers model	TCP/IP Model
Application layer	AFL – Application functions level	Application layer
	ADL – Application data level	
Network access layer	RTL – Routing level	Transport layer
	TDL – Transport data level	Network layer
	BRL – Bridging level	
Hardware layer	DLL – Data link level	Data link layer
	HAL – Hardware abstraction	Physical layer
	HDL – Hardware driver level	
	HIL – Hardware interface level	
HPL – Hardware physical level		

Table 1—Check out the comparison between the proposed thin-level layer protocol and the best-known TCP/IP level layers.

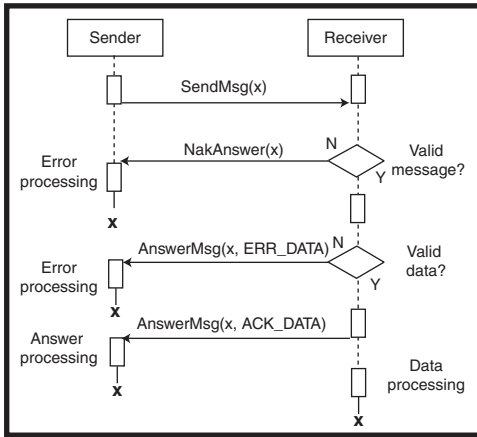


Figure 1—This is the communication flow between two devices and all of the possible types of answers.

protocols, but you can still implement a custom solution or interface your existing RS-232 protocol to TCP to provide the TCP/IP stack with a specific data link layer. So, it might be a good idea to design an application data protocol that could be used with existing protocols.

THIN-LAYER DESIGN

Let's say you want to design a versatile application data protocol that can be used independently by underlying protocols (e.g., TCP, UDP, CANOpen, and DeviceNet). You'll want it to be highly portable across different hardware architectures.

It's a good idea to think of the layers as bricks that fit one on top of the other. Changing a brick doesn't require you to change the others. Thus, you can create a homogeneous network or a heterogeneous one by changing some bricks. Table 1 is a comparison between the TCP/IP layered protocol stack and a thin layered protocol (TLM).

To use an application data protocol, you need about one brick for every layer model from the application to hardware layers. You can design a brick or it can be a part of the underlying protocol. Thus, it's possible to design a homogeneous network based on RS-485 hardware that provides an HDL (architecture-dependent), HAL, DLL, ADL, and AFL thin layers, or a TCP/IP network that provides AFL and ADL thin layers and uses underlying TCP/IP layers. Of course, to support more underlying protocols, the ADL and AFL protocol designs have to be more accurate. The main areas to investigate are

communication flow management and node addressing management.

NODE ADDRESSING

A node address in TCP and UDP has its own IP address. CAN has an 11- or 29-bit address. I²C has a 7- or 10-bit address. RS-485 and RS-422 don't have a standard addressing scheme, with the possible exception of the "ninth bit" addressing mode. The ninth bit in each byte is used to distinguish between address bytes and data bytes.

In every addressing scheme, all of the addresses have to be unique. They may be related to the node (physical devices) or to the data. (A data-sharing abstraction model is commonly used in CAN automotive networks.) When designing a heterogeneous network, it could be easier to provide every node with its own address (chosen independently from the underlying protocol) and to manage it at the network layer. A good solution is to use an IPv6 or a MAC address as a unique node address also in non-Ethernet networks. The Maxim Integrated Products DS2502-E64 1-Wire serial number chip contains a unique IPv6 address that can be used directly (or you can extract a MAC address from it). In any case, it's a good solution for low-volume production because it's less expensive than buying an IEEE number. I prefer to write a unique node address for every device. Doing so enables me to manage my device and download firmware upgrades via the network connection.

HIGHEST LAYER DESIGN

To define the application layer API, you must decide if you want a message-oriented or data-oriented protocol. Let's take a look at both.

Message-oriented protocols are commonly used in simple serial communications. (With ASCII encoded data, a typical C implementation is made with `printf()`-like functions to encode the entire command or answer with just one line of code.) They are the only possibility for interfacing

simple terminals. A message-oriented protocol may have a lot of specialized commands with specific parameters. But there are a few drawbacks. One is the computing time associated with encoding and decoding data. Another is that you have to modify the communication routines on both sides (sender and receiver) for new data.

A data-oriented protocol has a shorter list of messages: one for every type of data to transfer and some control messages. In the simplest scenario, your application will need only a few messages to read and write data. Because data-oriented protocols aren't usable with terminals, it is possible to pass the payload data in binary form without losing time to encode/decode them. It is also possible to implement simple remote procedure calls (RPCs) and deal with events (intended as a mechanism to signal asynchronous situations in a common way for every node). The main drawbacks associated with this protocol are data alignment (for record/struct data) and data Endianness management.

COMMUNICATION FLOW

Creating a network architecture involves defining how many devices (network nodes) will be communicating, determining how to connect the nodes, choosing the relationship between the nodes, and defining the rules for exchanging data between the nodes. PC networks are designed using the client/server architecture. Many industrial networks use master/slave and multimaster architectures, while others like CAN are implemented using a data-shared architecture.

Figure 1 shows a typical high-level

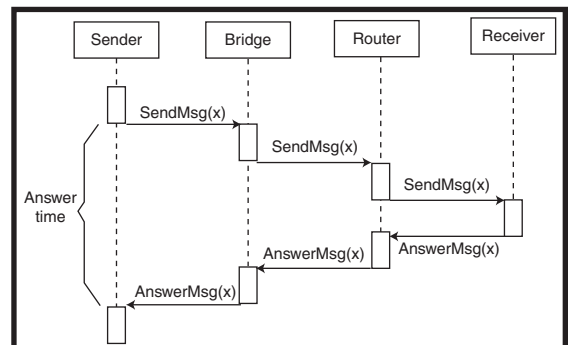


Figure 2—Communication messages route between two devices in a complex, heterogeneous network that is divided into subnetworks. The network can connect low-end devices and PCs sharing their data.

description for a message-passing scheme between sender and receiver devices. (I intentionally didn't mention master and slave devices.) It depicts a network without subnetworks and where every device has the same physical interface to the network, the more common situation in embedded networks. In Figure 2, the answer time is a function of the computing time in the sender, receiver, and bridge nodes added to every retransmission at the router level. The answer time can be several seconds long. The no answer timeout has to be bigger, so it is necessary to organize the firmware in a multitasking fashion (or at least using finite state machines, protothreads, or threads), especially in the sender devices. What can you do if the answer time is longer than what the application requires? You may write better software or, not surprisingly, find ready solutions compatible with your applications.

COMMUNICATION SOFTWARE

To write better software for the answer time, you must reduce the internal processing time spent for communication functions and you must reserve some processing time for them. Sender and receiver devices have to perform different activities (see Listing 1).

The receiver's activities are always the same (only the answer policy may change). In fact, the sender dictates the communication rules influencing data throughput. These functions are typically realized by data link layer (DLL) protocols. Using a higher layer protocol doesn't require you to deal with these problems; otherwise, they can be solved with DLL techniques. The simplest and least efficient is the stop and wait timeout scheme in which the sender waits to receive the answer to the previous command before sending the next one or a timeout occurs before error management processing. The receiver has to be able to distinguish if it has already

Listing 1—The sender makes a frame with data, sends it, and waits for an event that is usually the reception of the answer sent by the slave or a timeout fired for a missing answer. The receiver function is simpler because it makes the job in response to the sender.

```

void sender(PACKET *buf)
{
    FRAME s;
    EVENT evt=NONE_EVT;

    // make a frame with data to send
    FrameMake(&s, &buf);
    // ask HAL to send message
    HalSendMsg(&s);
    // wait answer
    EventWait(&evt);
}

void receiver(void)
{
    FRAME s;
    EVENT evt=NONE_EVT;
    ERROR err=ERROR_NONE;

    // wait for a message from sender
    EventWait(&evt);
    if(evt.type == EVT_MESSAGE_RXED) {
        // check if message ok
        err = MessageCheck(&s);
        //...if ok execute required actions
        if(ERROR_NONE == err)
            err = MessageExecuteActions(&s);
        // process error management
        if(ERROR_NONE == err)
            ErrorManagement(err);
        // send appropriate answer
        HalSendAnswer(&s, err);
    }
}

```

answered a message so every frame has a sequence number. Typical error processing involves a number of attempts at retransmission before a fatal error is generated. In this way, sender activities are the simplest, but they use more buffer memory and require acknowledgement from the receiver. In addition, error processing may be time consuming. More complex solutions are also more efficient. The sliding window approach and piggybacking are common ways of increasing data bandwidth.

HARDWARE-READY SOLUTIONS

There are two inexpensive hardware-ready solutions: CANbus and PC. Both can manage collisions using the CSMA/CA approach.

CAN is purely data-oriented. This means that data sent on the bus has its unique address and can be received by any device using an address filter-compare scheme. CAN also enables a receiver to ask for specific data using

the remote transmitting request scheme (RTR bit). A receiver can initiate a message transfer by sending an RTR message where the only interesting information is the address field (the address of the data it wants to receive) and an RTR bit set at a high logical level. The request and the answer are two completely different frames on the bus, so the answer may be delayed by other messages with high priorities, in addition to any delays caused by the application. An advantage of this feature is that the answer is not only received by the requesting receiver, but also by other possible receivers interested in this message, guaranteeing data consistency in the network that can be seen as a distributed application.

PC is a multimaster bus that consists of three lines: a bidirectional serial data line (SDA), a serial clock line (SCL: output for the sender, input for the receiver), and a common ground.

Address	R/W	Designation
0000-000	0	General call address (almost one slave has to acknowledge or stop condition will occur on the bus)
0000-000	1	Start byte
0000-001	x	Reserved for the (now obsolete) C-Bus format
0000-010	x	Reserved for a different bus format (obsolete, not used)
0000-011	x	Reserved for future purposes (not currently allowed)
0000-1xx	x	High-speed (up to 3.4 Mbps) mode master
1111-1xx	x	Reserved for future purposes
1111-0aa	x	10-bit Slave Addressing mode (aa: 2 MSbits, next byte contains 8 LSbits)

Table 2—These reserved I²C addresses shouldn't be used to assure compatibility with the I²C standard (except those for address 0).

Listing 2—Reading data from an I²C device is more difficult than writing it, and it is quite different from any other communication protocol.

Write sequence	Read sequence
<code>// send start sequence</code>	<code>// send start sequence</code>
<code>I2cStart();</code>	<code>I2cStart();</code>
<code>//send device address, R/W clear</code>	<code>// send device address,</code>
<code>I2cTx(0xE0);</code>	<code>// R/W: 0 to write, 1 to read</code>
<code>//send SFR08 command register address</code>	<code>I2cTx(0xE0);</code>
<code>I2cTx(0x00);</code>	<code>// send light sensor</code>
<code>// command to start ranging in cm</code>	<code>// register address</code>
<code>I2cTx(0x51);</code>	<code>I2cTx(0x01);</code>
<code>//send stop sequence</code>	<code>// send a restart sequence</code>
<code>I2cStop();</code>	<code>I2cStart();</code>
	<code>//send device address, R/W set</code>
	<code>I2cTx(0xE1);</code>
	<code>// get light sensor and send acknowledge.</code>
	<code>lightsensor = I2cRx(1);</code>
	<code>// note address auto increment feature</code>
	<code>//get range high byte and send ack</code>
	<code>rangehigh = I2cRx(1);</code>
	<code>//get range low byte and don't ack</code>
	<code>// last byte</code>
	<code>rangeLow = I2cRx(0);</code>
	<code>//send stop sequence</code>
	<code>I2cStop();</code>

I²C can be used at 100 kbps (low-speed mode), 400 kbps (fast-speed mode), or up to 3.4 Mbps (high-speed mode). It is used to interface a microcontroller to memories, peripherals, and other microcontroller-based devices. Its intrinsic ability to handle bus contention using a CSMA/CA approach and existing I²C bus buffers makes it attractive for applications targeted to RS-422/485.

As opposed to CAN, I²C addresses are related to devices. There are two address schemes using 7 or 10 address bits. Addresses between 0 and 7 are reserved, as you can see in Table 2.

The C code in Listing 2 is an example of a master writing and reading data to an SRF 08 ultrasonic sensor. The source code is posted on the *Circuit Cellar* FTP site.

DATA ENCODING

To send a data frame through a communication line, it is necessary to define a way to distinguish the beginning and the ending part of the frame using one or more special characters (see Figure 3). You also need a way to ensure that the frame is correctly received using a data checksum or a cyclic redundancy check (CRC).

Commonly used delimiting characters are DLE STX (start of frame) and DLE ETX (end of frame) ASCII characters. If you're like me and prefer to send payload data in binary form, it is

necessary to byte-stuff all DLE data and add one more DLE for every DLE character contained in the data. Table 3 is a list of byte-stuffing combinations.

Obviously, byte-stuffing has to be used only on the variable portion of the frame. As you can see in Figure 4, it is not mandatory to byte-stuff destination, source, sequence, and checksum fields if the frame's format is always the same, but it has to be used for the payload (the only field containing useful data).

ERROR DETECTION

Parity, checksums, and CRC can be used to detect errors in a datastream. Parity involves adding a bit to each transmitted byte to make the total number of ones odd (or even). It can detect all single-bit errors, but it performs poorly on multi-bit errors. A longitudinal parity check makes it possible to detect all errors on a single data byte, so simple parity checks aren't used in practice.

Adler-32 is the most notable summing checksum. It was designed to solve the problems associated with common parity and summing checksums. CRCs are more sophisticated error-detection algorithms. CRCs are used in different contexts,

but why are there so many of them? Is there such a thing as the perfect CRC? Are all CRCs useful? What is the best CRC for a specific application?

Take a look at Table 4, which shows the performance of CRC polynomials. It seems as though all CRCs perform well. An 8-bit CRC has a 99.21875% probability that an error will be found in the worst case. The probability rises to 99.997% for a 16-bit CRC. The conventional wisdom is that the best way to select a CRC polynomial is to use one that is commonly used. For this approach, it is assumed that those polynomials were selected for optimal error detection, which in some cases is incorrect. Remember that coding 2,048-bit data with a 16-bit CRC implies that a lot of data combinations produce the same CRC. For example, several standardized 16-bit polynomials have error-detection performance that's inferior to the available alternatives. Most books about coding theory present only a handful of published polynomials and provide little to no guidance about polynomial selection. In practice, you may distinguish a CRC polynomial by knowing its Hamming distance (HD) and its Hamming weight.

Note that a Hamming weight "N" is the number of errors (out of all possible message corruptions) that are undetected by a CRC using a particular polynomial. A set of Hamming weights captures performance for different numbers of bits corrupted in a message at a particular data word length, with each successively longer data word length having a set of Hamming weights with higher values. The first nonzero Hamming weight determines a code's Hamming distance.

The well-known CCITT-16 polynomial 0x8810 detects all 1-bit errors and all 2- and 3-bit errors for data words that are 48 bits long. (0x8810 is a hexadecimal representation of the polynomial $x^{16} + x^{12} + x^5 + 1$, with x^{16} being the highest bit and an

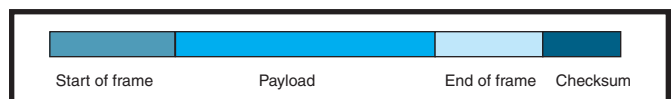


Figure 3—This is the format for a general frame with delimiters. The checksum/CRC field is appended at the end.

First character	Second character	Meaning
DLE	STX	Start of frame
DLE	ETX	End of frame
DLE	DLE	A byte of data containing DLE
DLE	Any other data	Impossible, there is an error on the frame

Table 3—Check out some byte-stuffing combinations for the ASCII characters used to delimit the frame.

implicit +1 term.) However, it provides only HD = 4 at this length because it fails to detect 84 of all the possible 4-bit errors. For the same data length, a CAN 15-bit polynomial 0x62CC—which is optimized for data word sizes up to 112 bits—provides HD = 6. The 12-bit polynomial 0x8F8 can achieve HD = 5, while the polynomial 0xC07 achieves only HD = 4. The 8-bit polynomial 0xEA also has HD = 4, and is better than the published DARC-8 polynomial, which does not. These examples point out three fundamental problems with current practices: there are gaps in the set of published polynomials; there is a need for guidance on which polynomials to use (and when to use them) and when you can't rely on CRC alone to protect your critical data.

There is a simple algorithm to follow for choosing a CRC polynomial: given that you are choosing a CRC polynomial p of degree g for a maximum data length of r bits, choose g so that 2^g is $> r$; choose p among prime polynomials of degree g that have at least two nonzero terms; multiply p by $(x + 1)$; and proof p against a set of bit sets whose lengths are $2p, 2p + 1, \dots, r^2$ bits to calculate its hamming distance.

Well-known polynomials are described in a 2004 article by Philip Koopman and Tridib Chakravarty titled "Cyclic Redundancy Code (CRC) Polynomial Selection for Embedded Networks." You will find a method for choosing a good CRC polynomial and C source code on the *Circuit Cellar* FTP site.

DATA FLOW & FRAME FORMAT

Can a sender transmit too much data for a receiver? Without data flow control it can. Simple data flow like XON/XOFF is inefficient. A protocol can require that every node has to accept a frame of a fixed maximum length. For example, a microcontroller can be required to accept a 32-byte

message and a PC, a 10-KB message.

In the master-slave architecture, the solution is fairly simple. The protocol can provide a mes-

sage to determine the maximum accepted data length (and other information related to the node). The master can make a table containing this data for every slave so it has to write the table just once during each power-up. It can update it every time a new slave node is added. In client/server architecture, the situation is similar. In the multimaster architecture, every node that can act as a master needs to work as the master in the master-slave architecture.

Major problems arise on a network implementing data-shared architectures. The CAN solution involves a frame with a maximum fixed-length message. Short messages may be a good option for reducing high-priority message latency time, but they result in poor bandwidth use. Long messages improve bandwidth utilization and increase the latency time of high-priority messages. So setting the maximum message length is a compromise.

Note that the packet frame is almost defined. Refer to Figure 3 and the complete data link level design. What data must be exchanged?

OBJECT-ORIENTED PROTOCOLS

If two processors could pass their shared data directly, the programming

technique would be similar to the one used when programming with a multi-tasking operating system. Data is generally in binary format. This means that problems regarding data Endianness and byte alignment have to be solved.

In practice, a big Endian processor and a little Endian processor will not read the same data (if it is multibyte) and they have to agree on a way to represent the same data. The common solution is to express all of the data in a frame in big Endian notation (or in little Endian notation). Alternatively, every packet can specify its Endianness (usually the Endianness of the transmitter node) with a bit inside a control character always present in every frame.

A good candidate would be the sequence character in Figure 4. The four lower-order bits may be the sequence number, and one of the other free bits may indicate if the frame is expressed in little Endian (0) or big Endian (1) notation. A more difficult decision would involve designing or using a simple high-level API to send and receive data. This would execute a remote function and signal and receive notification events with a simple syntax.

Figure 4 is a detailed frame format. The E flag indicates big Endian (1) or little Endian (0). The three free flags can be used for other protocol-specific needs.

A typical command set for a data/object-oriented protocol can include: obtaining device status, reading memory data, writing memory data, reading variables, writing variables, and executing functions. Obviously, variable-

Type of error	Error detection performance
Single bit error	100.00%
Double bit adjacent errors	100% as long as the generating polynomial has at least 3 nonzero terms
Any double bit adjacent errors	If the crc polynomial will not divide into any $(x^k + 1)$ for k up to the frame length, then all 2-bit errors will be detected.
Odd number of bit errors	100% as long as the generating polynomial is divisible by $x + 1$
Even number of bit errors	Polynomial and data length dependent
An error burst of length less than $r + 1$	100.00%
An error burst of length equal to $r + 1$	$E_{\%} = \left[1 - \left(\frac{1}{2} \right)^{(r-1)} \right] \times 100$
An error burst of length greater than $r + 1$	$E_{\%} = \left[1 - \left(\frac{1}{2} \right)^r \right] \times 100$

Table 4—This is a general CRC error-detection capability. r is the polynomial order.

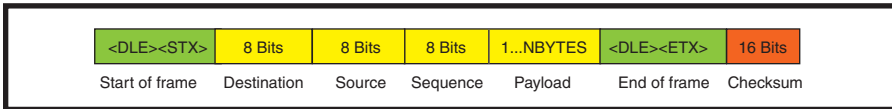


Figure 4—A general frame format details what is issued in a network protocol with a node-addressing mechanism.

related commands need to specify a type, length, and type of memory (RAM, ROM, EEPROM, etc.). Function-related commands need to specify I/O parameters, their types, and lengths.

Memory addresses can be used to distinguish among variables, but they may change if the source code is modified and recompiled. As a result, it is better to write a table containing the variables' characteristics and access them with an indexing mechanism. For flexibility, the protocol may need a way to define a record data type. Using the same philosophy as before, a type table can be used where every element describes a record/structure type with a syntax like the one used to specify data in `scanf/printf()` I/O functions. The big advantage is that with a reduced command set and two or three tables (for variables, functions, and types), it's possible to manage a big set of data and implement a simple remote procedure call mechanism.

DATA-ORIENTED PROTOCOLS

Most custom serial protocols are command oriented, while standard protocols tend to be data/object oriented. I have designed both types, but I prefer binary data-oriented protocols because they can work with a restricted set of messages and they don't require time-consuming conversions like ASCII protocols. Although it is common to pay little attention to error detection/correction methods, be sure to choose a CRC that addresses the application and data needs. ☒

Massimo Manca holds a certificate in Electronics Engineering and has studied Computer Science at the University of Udine. He owns Micron Engineering and his main interest is embedded software. He may be reached at massimo.manca@micronengineering.it and www.micronengineering.it.

PROJECT FILES

To download code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/208.

RESOURCES

P. Koopman and T. Chakravarty, "Cyclic

Redundancy Code (CRC) Polynomial Selection for Embedded Networks," Carnegie Mellon University, 2004, www.ece.cmu.edu/~koopman/roses/dsn04/koopman04_crc_poly_embedded.pdf.

LegacyJ Corp., "EBCDIC Table," 2006, www.legacyj.com/cobol/ebcdic.html.

SOURCE

DS2502-E64 Node address chip
Maxim Integrated Products, Inc.
www.maxim-ic.com

Professional Features – Exceptional Price

34 Channels sampled at 500 MHz

Sophisticated Multi-level Triggering

Transitional Sampling / Timing and State

Connect this indispensable tool to your PC's
USB 1.1 or 2.0 port and watch it pay for itself within hours!

- 500 MHz Sampling / Timing Mode (Internal clock)
- 200 MHz Sampling / State Mode (External clock)
- Multi-level Triggering on Edge, Pattern, Event Count, Group Magnitude/Range, Duration etc.
- Real-Time Hardware Sample Compression
- Qualified (Gated) State Mode Sampling
- Interpreters for I²C, SPI and RS232
- Integrated 300 MHz Frequency Counter
- +6V to -6V Adjustable Logic Threshold supports virtually all logic families
- Full version of software free to download
- Micror adapter available

www.pcTestInstruments.com

Visit our website for screenshots, specifications and to download the easy-to-use software.

Intronix Test Instruments, Inc.
Tel: (602) 493-0674 • Fax: (602) 493-2258
www.pcTestInstruments.com



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 **as low as \$99** qty 100
- Fanless, no heat sink
- SDRAM - up to 128MB **\$129** qty 1
- Flash - up to 128MB onboard
- 10/100 Ethernet - up to 2
- DIO lines - up to 55
- 2 USB ports
- COM ports- up to 10
- Programmable FPGAs
- Linux, Real Time extension, NetBSD
- SD card option
- VGA video

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



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

• options include:

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

133 MHz 586

- 5 boards in series **as low as \$229** qty 100
- Power as low as 800mA
- 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)

- Over 20 years in business
- Open Source Vision
- Never discontinued a product
- Engineers on Tech Support

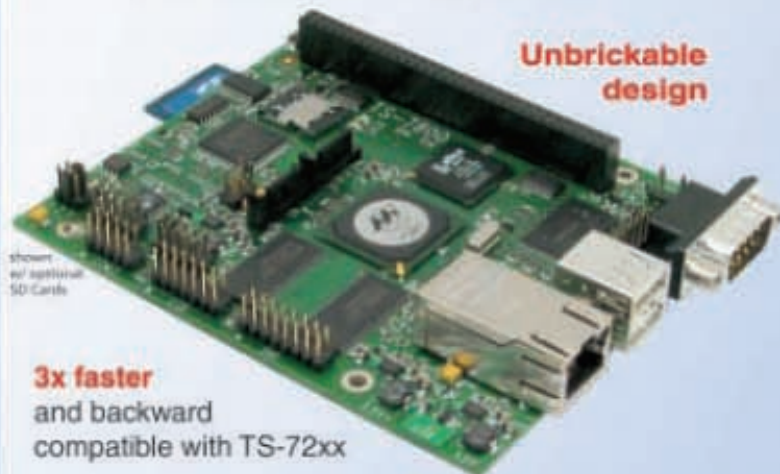
- Custom configurations and designs w/ excellent pricing and turn-around time
- Most products stocked and available for next day shipping

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

New Products and PC/104 Peripherals

NEW!

High-End Performance
with Embedded Ruggedness



shown
w/ optional
SD Cards

3x faster
and backward
compatible with TS-72xx

**Unbrickable
design**

TS-7800 500 MHz ARM9

- Low power - 4W@5V
 - 128MB DDR RAM
 - 512MB high-speed (17MB/sec) onboard Flash
 - 12,000 LUT user-programmable FPGA
 - Internal PCI Bus, PC/104 connector
 - 2 USB 2.0 480 Mbps
 - Gigabit ethernet
 - 10 serial ports
 - 5 10-bit ADC
 - Sleep mode uses 200 microamps
 - Boots Linux in < 2 seconds
 - Linux 2.6 and Debian by default
- \$229**
qty 100
- \$269**
qty 1

NEW!

TS-POE100 Power-Over-Ethernet

- 2.4 Amp 5V isolated power (12W)
- PC/104 peripheral w/ 10/100 Ethernet
- Sleep Mode & Wake-on-LAN with PoE



\$99
qty 1

board shown
with optional
16 pin PC/104
connector and
2x RS-485

NEW!

TS-PLC Programmable Logic Controller

- Use standalone or cluster via wireless or RS-485
- 7 analog/digital I/Os
- opt. 500 meter wireless
- Program in ladder logic or C
- Web interface
- 4.7-30VDC or USB power input
- Rugged aluminum enclosure 3.6" x 3.1" x 1.2"

\$89
qty 1



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

Resilience in Embedded Designs (Part 3)

Software

In this series of articles, Aubrey has been coaching you through the process of protecting your embedded designs. This month he concludes the series with some tips on software design for small systems.

Resilience is often attributed to protection in the hardware domain; however, adopting defensive techniques while writing software can result in a significant benefit. This article on approaches to software design is the third in my series on imbuing a design with resilience.

Because software is normally written to handle several tasks that can be asynchronous, the possibility of intermittent errors that do not show up during normal testing is enormous. Code is broken into segments and tested as such. Combining them into a single program requires a good overview of the objectives of the system and how its parts interact. In the description of the software, I will address only smaller systems. Large systems with associated teams of programmers and adherence to certain standards are the subject of many studies with resulting methodologies for implementation. Small systems don't normally have the memory resources to support the techniques.

TOOLS

When developing software, issues that have nothing to do with software may cause glitches in operation. If you use a real-time operating system (RTOS), how certain are you that it multitasks predictably and without error? Does the compiler compile without bugs, and if it doesn't, will your tests discover those bugs? Is your high-level language suitable? As a matter of interest, there is a specification from

the Motor Industry Software Reliability Association (MISRA), "Guidelines for the Use of the C Language in Vehicle Based Software," which describes the C constructs that are and are not allowed for safety-critical systems.

RTOS

Rather than deal with a specific RTOS (commercial or otherwise), I will first explain a cooperative multi-tasking system that is widely used in

simple systems. Concepts that are applicable to it may be extended in one form or another to more complex systems, including an RTOS.

A system can be broken into a number of tasks. Some tasks are independent while others have some degree of interrelationship. Where a relationship exists, some form of handshaking using flags and variables realizes communication between the tasks.

In Figure 1, the routine associated with each task is executed in sequence. It yields control to the next task when it is good and ready to. The decision over when the ideal point to break occurs is at the whim of the programmer. One of the advantages of the technique is that there is no need for context switching (the storing of multiple variables associated with each state) for each task. On the other hand, if not carefully coded, one task can hog the system resources so other tasks are not executed. It often helps to break each task into subtasks in a similar manner.

THE UNEXPECTED

A cautionary note for any time the switch command (or any action based on a selection) is used: always add the default to cater to the unspecified states. Whether you want to trap the error or perform a graceful recovery is up to you, but you will be surprised at the number of times the "impossible" combination occurs. Without the associated software, the system will crash without the handle to discover why.

The concept should be applied to each

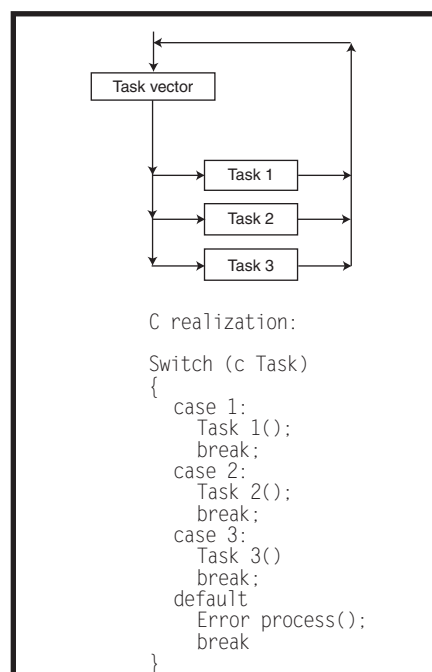


Figure 1—As each task completes, the process returns to the task vector mechanism, which directs it to the next task. It is possible that this vectoring process could maintain a record of which task is executed next, but I believe that it is more resilient to include the increment of the task pointer (cTask in this example) in each task, although this is not detailed here.

and every case in which an unforeseen error may creep in. Consider a scenario in which a 10-bit ADC reading is done as two 8-bit numbers. Even if the microcomputer's datasheet stipulates that the upper bits are zero, there may be circumstances in which this is not the case. Make sure the bits are zero, or you may find unpredictable and inexplicable errors. (True story!)

BACKGROUND/INTERRUPT

In addition to the "background" program execution, there is normally at least one interrupt action associated with a timer. As long as the interrupt occurs regularly, it is possible for the background to check that the interrupt is occurring and vice versa (see Figure 2).

Depending on the realization, there is a potential problem when `cCheck1` overflows back to 0, so make sure that your code can handle this instance or you will get false errors. It is a philosophical question of how to proceed once an error is detected, one that I will address a little later.

REGISTERS

There are many registers in a microcomputer that contain information ranging from variables to interrupt masks and even the program counter. A transient of some form (e.g., a power supply glitch, radiated noise, and alpha particle radiation) can upset any one of the multitude of bits. Depending on the register, the effect can be immediate (i.e., program counter) or it can take much longer to manifest itself.

PROGRAM COUNTER

If the program counter is garbled, the microcomputer starts executing code at an erroneous location. This can result in one of four things: It can go to code space where there is no code and execute whatever it finds there. It can go to data memory (if the architecture permits it) and start executing data as code. It can execute code out of phase so immediate bytes may appear as instructions or a similar misinterpretation of the code. Finally, it can start exe-

cuting code elsewhere, out of sequence, with the overall system operation.

When code is assembled or compiled, it normally ignores any memory location within the code space that is not used. The solution to the problem is to fill unused bytes with predetermined code. That forces the microcomputer into an error detection and correction mode if an attempt is made to execute code at an unused location. It is possible to fill the unused bytes with GOTOs, or no-ops,

and then a GOTO at the end of the block, or if your processor allows it, software interrupts. Whichever technique is used, it is preferable to use single-byte instructions because there is no guarantee which byte is accessed first. How you achieve this depends on a number of things: the assembler and the compiler, or if it is easy to fill the memory with a code pattern and then overwrite it with code. Sometimes the code you use may have even greater ramifications and you may need to think outside the constraints of regular code. In some microcomputers, code can change a setting and the setting can be changed back only by a hardware reset. If this is code (which is in fact "data"), it may not be possible to gracefully recover from a detected error. On larger or older processors, it is possible for the program counter to end up in external ROM space where no memory exists. It is possible to add hardware to generate a single byte-read instruction.

In the other three cases, there needs to be another technique to detect the program misstep. You need to decide what is likely to be out of step and use the mismatch to detect the fault because a hiccup has occurred. The first thing you can do is check the correct sequence of your tasks. For instance, you could add sample code in the tasks first described in Figure 1 (see Listing 1).

An additional suggestion, which is normally more difficult to implement in a high-level language, involves checking the level of the stack (where the microcomputer has this facility) at the start of the procedure (subroutine), at the end, or both. You could also pass parameters on calling a task and measure the returned value. In critical cases, you could even use pseudorandom data for the checks.

The checks can be done at any part of the process, but you need to consider what is reasonable for your application. One approach is to have a common exit point for all of the tasks where the stack level is investigated. The code above, and for all tasks, may look like Listing 2.

CHECK REGISTERS

If there is spare time in the execu-

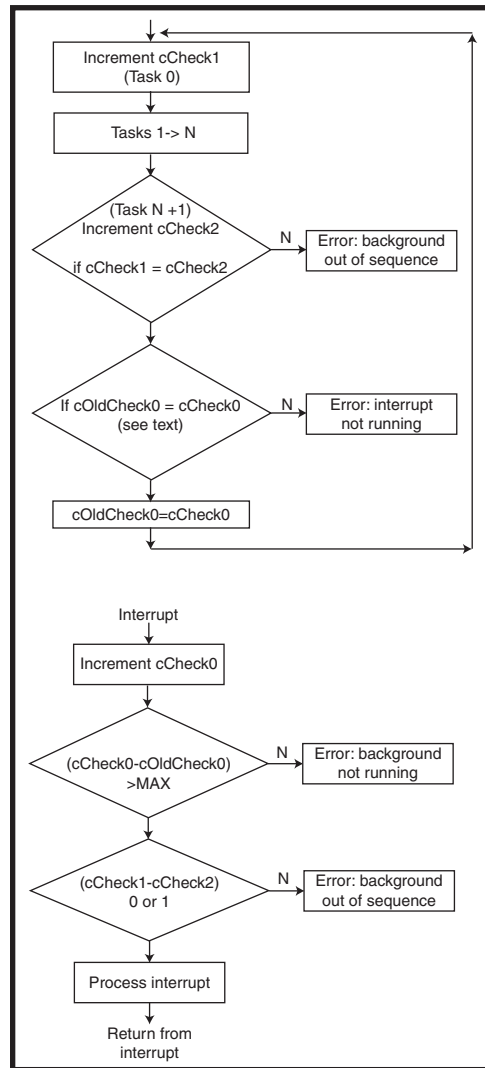


Figure 2—In this sequence of events, it is presumed that the periodic interrupt occurs at least once during a complete task cycle. During the interrupt process a counter, `cCheck0`, is incremented. When the background process compares this to a stored value, `cOldCheck0`, it must have changed or the interrupt process is not running. In addition, there is a check in the interrupt process that the difference between the two variables is less than a maximum number, which effectively checks the maximum number of interrupts that can occur during a task cycle and may need to be fine-tuned empirically. In the background sequence, `cCheck1` and `cCheck2` are incremented once in a complete task cycle. There can be a difference of only 0 or 1 between them, depending on when an interrupt occurs. If this is not the case, the background process must be out of sequence.

tion cycle, you can compare the register settings with the supposed values. A handy way to do this is to create a table with all the settings in ROM, which is used for both set up and verification. Unfortunately, some microcomputers have write-only registers, so the registers cannot be tested directly. Sometimes the register contents can be checked by forcing an event and monitoring the results, but I suspect that if you are going to have to do this, perhaps you should use another microcomputer.

RAM

Any volatile memory location can be changed by the same mechanism as the registers. Several techniques can be applied to detect the upsets. As a simple approach, dedicate several locations as check bytes and write values to them. If an error is detected, take the appropriate recovery steps, presuming that other locations have also been changed. The shortcoming of the simplistic approach (especially on its own) is that there is no test done on locations that are critical to software operation.

It is also possible to use redundant locations for critical variables, and if they disagree, an error is noted. Perhaps the most practical method is to group the critical variables and generate a checksum or a cyclic redundancy check (CRC). Obviously, the validation value has to be updated any time one of the variables changes. This can prove to be a time-hogging process. By using a checksum approach, rather than a CRC, subtracting the old value, and adding the new one can reduce the time needed.

The actual check doesn't have to happen every time through the tasks. It can be executed every second or whenever there is spare time available.

GRACEFUL RECOVERY

When an error is detected, there are several approaches that can be taken to recover. The most certain is to execute a tight loop without clocking the watchdog. When the watchdog times out, the system is completely reset. Doing this may lose some measurements or calibration settings, especially if you have coded in C and all the variables are initiated to zero on reset. In addition, part of

Listing 1—At the start of every task, the software checks to ensure that the task pointer is correct for the task. This way, if the microcomputer is executing code out of sequence, there is a good probability that the two will not match and an error-correction process can be invoked.

```
Task1 ()
{
    if cTask!=1
    {
        ErrorProcess();
    }
    else {
        //other code
    }
}
```

the error handler may record a code associated with the error to help interpret the fault. A reset may also cause the I/O lines to change back to their original states, generating unwanted outputs.

Some microcomputers provide flags to indicate what caused the reset condition, but even without this feature, a possible way to address the issue is to have a "reserved" block that is not initialized under all circumstances. It could be protected by a CRC (or checksum) to determine if the data is valid. A check of the consistency of the CRC with the memory contents could be done at every reset. If the CRC does not check out, a complete initialization is performed as opposed to a partial initialization, which leaves critical information untouched.

Alternatively, you can create a software reset by jumping to the reset vector location rather than enabling the hardware reset to do this. There are two areas of caution. First, you must ensure that the stack is always set to the correct location, either as part of the error handler or as part of the microcomputer initialization. Second, some actions taken by a microcomputer can be reversed only by a hardware reset. For instance, in some microcomputers, only the watchdog timer can be enabled. During the set-up process, you may not want the watchdog enabled while EEPROM parameters are set up.

A third alternative is to write a complete error handler that corrects the error and then returns to normal execution of the program. This requires great insight into the interrelationships between the hardware and software processes that occur in the project.

These considerations are the same

that must be taken when using the brownout detection that is available on some microcomputers.

WATCHDOG

I spent quite a bit of time discussing the hardware aspects of the watchdog last month. All of the considerations for the watchdog can be invalidated by compromising the design in software. Ideally, the watchdog timer should be clocked by at least two actions. One action should occur in one part of the program and the second in a different and preferably unrelated part of the program. If the program execution locks up in one part, it should not see the other so the reset will occur. For instance, in the example in Figure 2, the action to set the output line that resets the watchdog could happen as part of the background loop and the action to clear the output line could be placed in the interrupt process.

You can add additional checks for additional security, especially if you have a "weak" hardware watchdog. You can add a check to ensure each task is executing in the correct sequence by creating a CRC-type shift register that is clocked in every task. The final task checks to see if the value in the shift register is correct, if the interrupt process is occurring (plus any other check), and only then clocks the watchdog.

Another approach is to add software watchdog timers to each task. Each timer must be refreshed each time the task is executed, and if any one times out (as monitored in a separate process), the watchdog is activated.

It should not be possible to clock the watchdog from two different places in the software because it may prevent the watchdog from triggering in a lockup

Listing 2—When using the software sequence described in Figure 1, each task executes at the same stack level. If the software-execution sequence is upset by an external event and one of the tasks is completed, there is a possibility that the stack level would uncover the upset. To reduce code, it is possible to have a common return point that checks the stack level, as shown here. Unfortunately, it is not possible to check the stack level on some microcomputers and it may also be difficult to code in high-level languages.

```
Task1 ()
{
    if cTask!=1
    {
        ErrorProcess();
    }
    else {
        //other code
    }
    goto StackCheck
} //never actually returns from here

StackCheck:
;in assembler
;'check stack level
    ret
;or jump depending on the implementation.
```

condition. However, this creates a quandary with internal EEPROM or flash memory programming because the microcomputer can actually go into a wait state while this is happening and the microcomputer can time out. There are no hard-and-fast solutions that I

know about. Just remember to consider what will happen in your set of circumstances.

I always test my designs with the actual reset turned off so I know when the watchdog triggers. Well-designed watchdogs allow for seamless operation

so it may be beneficial to have some kind of visual indication that the event happened to make sure there isn't some latent problem that recurs.

ADDITIONAL CHECKS

There are often secondary processors in the system. They include A/D subsystems and display controllers. The software should be capable of detecting a subsystem failure and then reinitiating the subprocessor. In the case of the Hitachi HD44780 controller used in most character LCDs, the check could be for the busy flag (BF) bit, and if it continued to be busy for longer than a fixed time period, the host would attempt to reset it through software commands and failing that by cycling the power to the device. I have seen descriptions of how to convert the interface to the display to a two-wire interface with shift registers. Using the serial approach converts the display to "write only" and the system has no idea if the display is operational.

Always check communications loops for correct operation. If nothing

Data Acquisition & Control Computer

iPac 9302

- Cirrus Logic EP9302 ARM9 200 Mhz Processor
- Floating Point Math Engine
- 2 USB 2.0 Host Ports
- SD/MMC Flash Disk Slot
- 48 Digital GPIO Lines
- 1 10/100 Base-T Ethernet port
- 5 channels of 12 bit A/D & 3 PWMs
- 1 RS232 & 1 RS232/422/485 Serial Port
- Battery Backed Real Time clock/calendar
- Eclipse Linux Development Environment






2.6 Kernel
net Micro Framework

The iPac has enough I/O for demanding applications & with a size of 3.5" x 3.8" it can fit almost anywhere. Prices start at \$150.00. Please contact us for more information.

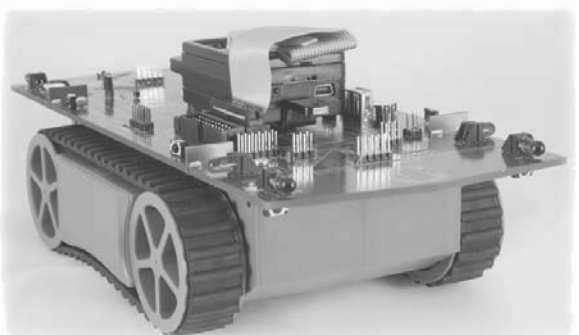
Since 1985
OVER
22
YEARS OF
SINGLE BOARD
SOLUTIONS

EMAC, inc.

EQUIPMENT MONITOR AND CONTROL

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

TrackBot™



shown with optional SunSPOT (purchase separately)

TrackBot is an affordable, autonomous robot platform with a real-time P2P network of RISC chips which manage all transducers (including custom ones you can add). Any controller with a standard UART can be added as the high-level application brain. It's lead-free and RoHS-compliant. TrackBot was developed specifically for university-level education and research. It's so unique that multiple patents are pending. TrackBot is in stock and shipping now!

SYSTRONIX®

939 Edison Street, Salt Lake City, Utah, USA
www.trackbot.systronix.com

is detected for a predetermined period, action should be taken to ensure that the system continues to function properly.

Make sure that analog current loops or voltage inputs are functional by ensuring that the readings are legitimate and remembering to cater to the case where they are outside the permissible range, even though you or your customer are convinced it could never happen.

INTERRUPTS

One of the most common causes of unpredictable problematic behavior in firmware operation is the action and interaction of interrupts. The problems can be generated by several mechanisms: The stack can be too small or the time taken to save variables to the stack blinds the processor to critical input changes. Nested interrupts can exacerbate a problem. The execution time of an interrupt can take too long and a subsequent interrupt (of the same or different type) can be missed. There can be a problem during the read-modify-write of a register or memory location when an interrupt occurs during the process. This can be a particularly difficult problem to track down, especially when applied to I/O ports. On processors with simple I/O configurations, this can often lead to an input being inadvertently disabled. Careful consideration of the action of the interrupts can save hours of debug time.

HARD FAILURES

Most hardware failures can be monitored by the microcomputer reading back outputs and comparing them to known inputs. It can also be done with redundancy or majority voting systems. It all depends on how critical your mission is and how you are going to handle the error, but this strays into the hardware realm. What does remain in the software domain for some applications is checking that the microprocessor instruction set is executing correctly. Isolating a subset of the instructions to execute and checking that they have executed correctly is no small task.

TEST, TEST, & TEST

There is no substitute for extensive testing in an environment that resembles

the real world. Try to consider every possible (and impossible) set of events. Even if you have been told that something physically cannot happen, try to see how your system reacts when it does happen. In many applications, there are long-term timers that last weeks, months, or even years, and they are really difficult to check in real time. Add a software or hardware setting that enables you to accelerate time and modify the accumulated values so you get to see the transitions at critical junctures.

Peer review will frequently highlight oversights or challenge presumptions, even if it doesn't involve a formal code review. Ask a coworker, your boss, or a friend to try to operate the system. Even pressing buttons aimlessly can provide your system with an unexpected workout.

STAY SAFE

This has been a difficult article for me to write. It touches on many aspects of a design and it has been hard to decide which is good design practice and which is pertinent to resilience. Also, I hasten to add that unlike most of my other publications, very little of this is original. It's just that I have accumulated a lot of information over the years. I hope that it provides some insight on designing systems that continue to work for a long time.

I have touched on many ideas. Unfortunately, there is not enough magazine space to discuss them fully. Most of the material is available on the Internet. I have provided a separate file, ResilienceReferences.pdf, which you may download from the *Circuit Cellar* FTP site. ☒

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

PROJECT FILES

To download ResilienceReferences.pdf, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/208.

RESOURCES

N. Burnham, "Fault-Tolerant Software in Real-Time Single-Chip Microcontroller Systems," *Electronic Components and Applications*, vol. 6, no. 1, 1984.

J. Ganssle, "Born to Fail," *Embedded Systems Programming*, December 2002, www.embedded.com/story/OEG20021211S0032.

———, "Li'l Bow Wow," *Embedded Systems Programming*, January 2003, www.embedded.com/story/OEG20030115S0042.

———, "MISRA Minimizes Mishaps," *Embedded Systems Programming*, November 2006, www.embedded.com/shared/printableArticle.jhtml?articleID=193500806.

———, "Watching the Watchdog," *Embedded Systems Programming*, February 2003, www.embedded.com/story/OEG20030220S0037.

D. Jarrett, "Software Fault Tolerance Staves off the Errors That Besiege μ P Systems," *Electronic Design*, 1984.

Motor Industry Software Reliability Association, "Guidelines for the Use of the C Language in Vehicle Based Software," www.misra.org.uk.

G. Novacek, "Fault-Tolerant Electronic Systems," *Circuit Cellar* 162, 2004.

T. Williamson, "Designing Microcontroller Systems for Electrically Noisy Environments," AP-125, Intel Corp., 1993, <http://download.intel.com/design/auto/mcs96/applnots/21031302.PDF>.

B. Yarkoni and J. Wharton, "Designing Reliable Software for Automotive Applications," AR-102, Intel Corp., 1979.

SOURCE

HD44780 LCD Controller
Hitachi
www.hitachi.com

SMT Manufacturing

Take a Board from Prototype to Production

Zack walks you through the full development cycle of a USB-to-serial adapter for a programmable knob. The design is intended for car computer systems where it can be programmed to support multiple functions. Now you too can take a similar design from prototype to production.

If you've never taken a circuit board from prototype through manufacture, or if as part of a larger development organization some parts of the process are a mystery to you, you're not only missing some of the fun, you're missing valuable knowledge. In this article, I will cover the full development cycle of a small project from conception through manufacture, with an emphasis on how to deal with outside vendors to produce a professional product.

DESCRIPTION

The project covered in this article is a USB-to-serial adapter for a programmable knob, also known as a haptic controller. (Haptic means "touch." A haptic knob electromechanically reacts to your finger. You might be familiar with video game joysticks that vibrate when you fire. The vibration is an example of a haptic effect.) The knob is intended to be used in automotive computer systems where it can be dynamically programmed via software to support multiple functions. For example, while viewing a

navigation map, the knob may function as a zoom with 10 "click" points and restrict the turning range to the maximum and minimum zoom. When the driver subsequently changes to the MP3 audio menu, the knob should provide an infinite number of "clicks" so he can scroll through his song list. Finally, when the driver selects a song, the same knob can act as a volume knob with a smooth feel that becomes slightly stiffer as the volume increases. I used an Immersion PR-1000 haptic

knob that exposes a serial interface and operates from a 5-V, 500-mA power supply. I chose to control it from my in-car computer via USB.

The inexpensive board is small enough to fit in any nook of a dashboard. Four pins are exposed: 5-V (500-mA), ground, and two TTL serial lines. It is powered and controlled from a Mini-USB plug (smaller than a standard USB), and it functions within a vehicle's temperature environment (-20° to 80°C). Also note that the board

complies with the USB specification for standby, which is especially critical in a car where a power leak can drain your battery. Finally, note that the board is RoHS-compliant. Often sloppily referred to as lead-free, a board is RoHS-compliant if no parts contain more than the set maximum levels of lead, cadmium, and so on. European law requires all new electronic products to be RoHS-compliant.

DESIGN CAPTURE

The most important factor in producing an electronic board is the design. Design cap-

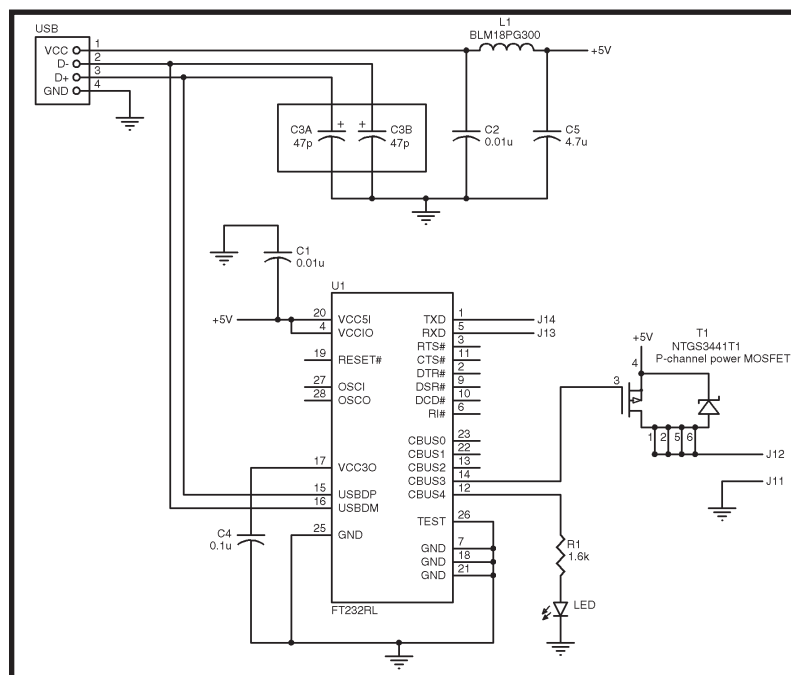


Figure 1—This is a circuit schematic for the PR-1000 USB interface board. Work your circuit over thoroughly before laying out your board. While modern CAD programs enable you to cycle between your schematic and layout, each change in the schematic can trigger a number of changes in the layout, especially on dense boards.

ture is the process by which you produce a schematic and a board layout, choose parts, and so on. At the end, you have computer files that can be sent to manufacturers to produce the PCB.

Even before you start to lay out your schematic in detail, your first order of business is to choose your parts. This can be a very involved process because choosing the wrong part can be deadly. Nothing is more frustrating than finding out that your chip is not RoHS-compliant or that it is insufficiently stocked to fill your manufacturing order before your Christmas season deadline in October.

Include all parameters when choosing your parts. Consider the following. Is your part generic? A complete resistor specification usually consists of a physical dimension, a resistance, and a maximum wattage. Is your part in supply? Is your part RoHS-compatible? Does your part restrict your soldering options? This is especially a concern with RoHS parts because a higher temperature must be used to melt the

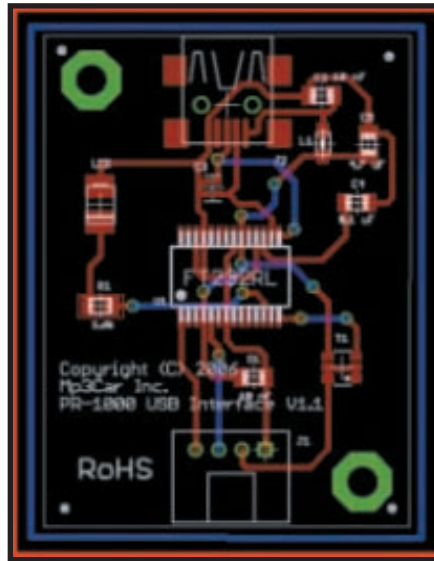


Photo 1—This is a board layout for the PR-1000 USB interface board. Red represents the top layer and blue is the bottom layer of copper on the final board. Because there are two layers of copper, this is called a two-layer board. (Source: MP3Car, Inc., www.mp3car.com)

solder on RoHS boards. Mechanical characteristics are important with plugs and sockets because they are physically stressed over the lifetime of a circuit board. Heat dissipation can

also be a concern. Be sure to note if your part is going to need heatsinks.

As you choose your parts, create a bill of materials (BOM) that lists the names of the parts and any appropriate data about them. The BOM is usually an Excel file and should contain several important pieces of information: the label of the part in the schematic (like R1, C1, JP4, etc.), the value of the part (i.e., “4.7k” for a resistor), the manufacturer’s part number (e.g., “CY8C20334-12LKXI”), and the manufacturer’s name (e.g., “Cypress”). In addition, your BOM may also include information such as links to datasheets, supplier names and costs, and package types (i.e., “0805” or “SSOP-28”).

When outsourcing component purchasing to the manufacturer (turnkey assembly), they will sometimes ask for an approved vendor list (AVL). The AVL enables the manufacturer to select a part compatible with your design. Most commonly, the AVL is just a pair of columns in your BOM that specify the manufacturer and part

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

FlashPro430 GangPro430



USB Flash Programmers for Texas Instruments' MSP430 microcontrollers.

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

- ✓ 60 kB Flash can be programmed in about 2 seconds
- ✓ supports JTAG, Spy-Bi-Wire and BSL interfaces
- ✓ can assign unique serial numbers
- ✓ up to eight programmers can be connected to one PC and program target devices simultaneously

New: FlashPro-CC and GangPro-CC

USB Flash Programmers for CC-series devices (ChipFon) from TI





www.elprotronic.com

USBee DX

Oscilloscope/Logic Analyzer

2 Analog and 16 Digital Signals
Up to 24Mps, 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 24Mps
Playback of Logic Analyzer Traces

Pulse Width Modulator

16 User Controlled PWM Channels

Frequency Counter

16 Channel Counter to 24MHz

Frequency Generator

Generates Sets of Common Frequencies

I2C Controller

Control I2C Devices Using Simple Text Scripts

Remote Controller

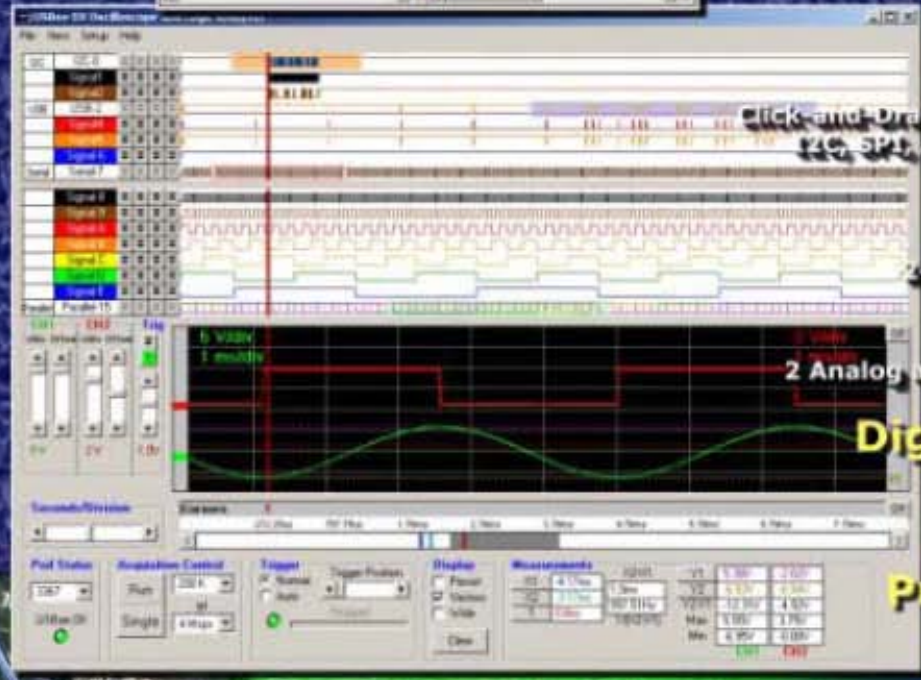
Easily Control Your Hardware Using Your PC

Pulse Counter

16 Channel Pulse Counter With Gating Control

plus the USBee Toolbuilder Source code and Library

Create Your Own Applications to Control the USBee DX



Actual Size

Data Extractors

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

Continuous Real-Time Embedded Bus Data Streaming

Store Bus Data to Disk or Send to Your Custom Application

Capture and Process Entire Test Sequences

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

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



number. If your part is generic, you can place “any vendor” under the manufacturer name, but if you do that, specify all the electrical parameters that matter! Otherwise, don’t whine when they put a generic resistor with a parasitic capacitance into your high-frequency circuit and it kills your board.

Keep track of the parts’ prices compared to the quantities needed. Even a difference of \$0.01 per part results in a \$1,000 difference over 100,000 boards.

SCHEMATIC CAPTURE

Schematic capture is the process of drawing a circuit schematic. You will need to choose a software package that suits your personality or work group. Unless you are an old pro, one of the best starting packages is Cadsoft Computer’s Eagle. Eagle is free for very small projects. A full version costs

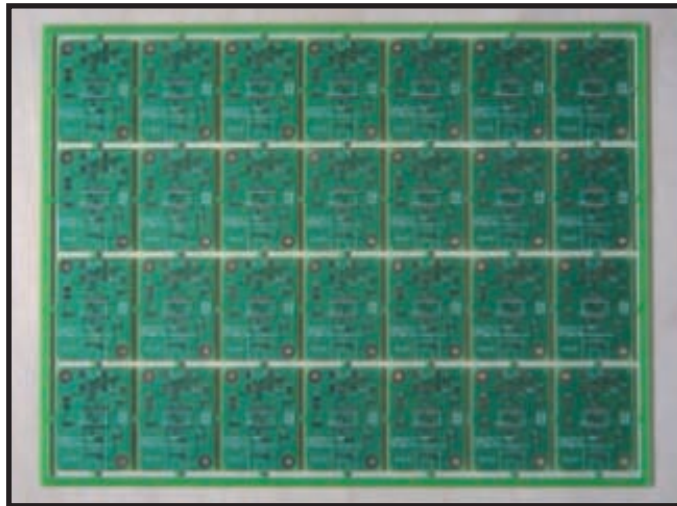


Photo 2—This is a panel containing 28 boards. The panel reduces your manufacturing cost because a robot can place parts and solder on multiple boards simultaneously when they are panelized.

about \$1,000. Other packages you might want to consider are Cadence Design Systems’s OrCAD and Mentor Graphics’s PADS, which can cost several thousands of dollars each. There are many professional software packages available, including specialized packages for high-frequency, digital simulation, and so on. In case

you don’t already know, these are called electronic design automation (EDA) packages. My circuit in Figure 1 was produced with Eagle.

The schematic produced during this stage is not just for you. When testing your assembled board, manufacturers will often ask for a copy of your schematic and netlist so they can test the electrical connections. A netlist is a file that contains a list of all nets (conductors at the same voltage) and all the connections to each net by components.

For example, ground is one net and the wire connecting a capacitor to a chip is another net.

The makers of high-quality design software provide good tutorials, so I won’t write about using EDA software. Nevertheless, you should realize that schematic capture is typically not a process independent of board layout or prototyping. Typically, you iteratively draw your schematic, prototype your circuit, and lay out the circuit board until your design is perfect. A good software package will integrate schematic capture, board layout, and simulation.

BOARD LAYOUT

Next, you must lay out the PCB. In addition to the tutorial that should come with your software, an excellent generic tutorial is available at alter-natezone.com. It covers the basics of PCB design, including a number of tips that make it a valuable read even for a semiexperienced designer.^[1]

Consider the box (or enclosure) that will hold your board. You need to provide enough large, correctly positioned holes on your board to easily accommodate screws and the shape of your box. You should also provide a metal surface around each screw hole that is at least as large as the screw head so it doesn’t interfere with the rest of your board.

Also, consider the position of your connectors with respect to the box. For example, while both of the PR-1000 board’s connectors are flush with

ARM7 • ARM9 • ARM11 • Cortex DaVinci • DSP • XScale • Marvell PXA

SW Development Tools

JTAGjet In-Circuit Debuggers

- ARM, XScale, MPCore, OMAP, TMS470, DSP TMS320, DaVinci & Marvell PXA device support
- Includes Chameleon Debugger for ARM
- High-speed (480 Mbps) USB 2.0 port
- Support for all major debuggers & C compilers

JTAGjet-Trace Debuggers

- ARM & TI OMAP/DM support
- Up to 4M deep real-time trace
- PC and variable tracing
- Up to 400 MHz
- Cycle accurate time

SIGNUM SYSTEMS

1211 Flynn Rd, Unit 104, Camarillo, CA 93012 (800) 838-8012 • www.signum.com

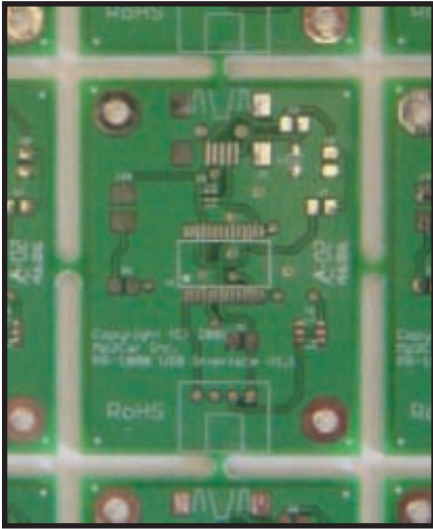


Photo 3—This is a single board still in the panel. The four tabs are not perforated. Be sure to include space between boards for overhanging parts, if necessary. (Source: MP3Car, Inc., www.mp3car.com)

the edge of the board, it is not always the case in other projects. Some connectors need to project beyond the edge of your PCB so they are flush with the edge of the outer box. Others require a minimum clearance to avoid hitting the lid of the box. Be smart. Design your board and box simultaneously (see Photo 1).

With the above in mind, follow this sequence when laying out the board: Consider the physical dimensions of your board and make it match the box or enclosure. Consider the position of any screw holes and external interface components (i.e., sockets, displays, buttons, and so on). Next, position complex parts first. They typically have more wires and require repositioning smaller parts to accommodate them. Finally, consider any heatsinks or other bulky components that must be added.

There are some common gotchas that can be easily avoided if you think about them. Include your copyright information, your product's name, and the version number in the silkscreen. Also include "RoHS" if your product is RoHS-compliant and any Underwriters Laboratories (UL) logos that apply to your design.

Always double-check your footprints. (The footprint is the shape that gets placed on the PCB for your component. It consists of pads that your component pins can be soldered

against, as well as a silkscreen describing the part label, etc.)

Manufacturing processes differ, so don't assume that your EDA software's ready-made footprint is the right footprint. Be ready to make a number of footprints by hand. Additionally, include fiducials on your board, which can be readily seen by robots. A fiducial is a mark used to define a coordinate system. It is usually just a small circle. They are used during manufacturing to place parts on your boards. Make sure your silkscreen doesn't overlap any other exposed layers. Often the silkscreen will look fine in your CAD program, but letters or symbols will be truncated because they were on top of a solder pad and you didn't notice. Include polarities of any polar devices. One manufacturer mentioned it was a common error to solder LEDs on backwards because the polarity marker was missing. I experienced this when my LEDs were soldered on backwards because my polarity marker overlapped a solder pad and wasn't visible! Finally, most manufacturing processes have a tolerance of a few thousandths of an inch. Don't push the envelope unless you have to.

MANUFACTURE & FABRICATION

To transform your design into a product, you will need a fabrication house that will produce the PCB itself and an assembly house that will solder

the components onto your board, which is sometimes called "stuffing the board." Many assembly houses will also do a "box build." They'll place your board in a box and connect it to cables, buttons, and so on.

Manufacturing the PCB is perhaps the easiest step because the hardware industry is board- and chip-centric. Finding the right board house is crucial. I used Advanced Circuits. It was professional, competent, and reasonably priced.

When choosing a fabrication house, make sure that you look at the quality of each company's boards. You'll learn this point from experience. Most fabrication houses will do panel layout. (Panels will be covered shortly.) Can you order a larger number of panels for the production run? For quantities from 10,000 to 100,000-plus boards, using a company with a facility in China might be necessary to remain competitive if you are in a saturated market. Make sure its tech people are quick to respond to your questions with answers you understand, and ensure it has all the technologies you will need. If you are doing high-frequency work, you may need to choose from a selection of board materials—not just the standard FR-4. (FR-4 is a fiberglass and epoxy resin board. FR stands for fire retardant. It is the most common board material.) If you are making a PCI board (or a similar one), you will need



Photo 4—Here is a reel of tape with resistors, a tube containing microchips, and a bag of bulk connectors. The tube and reel are standard sizes and can be fed into most assembly machines. The bulk connectors will have to be placed by hand.

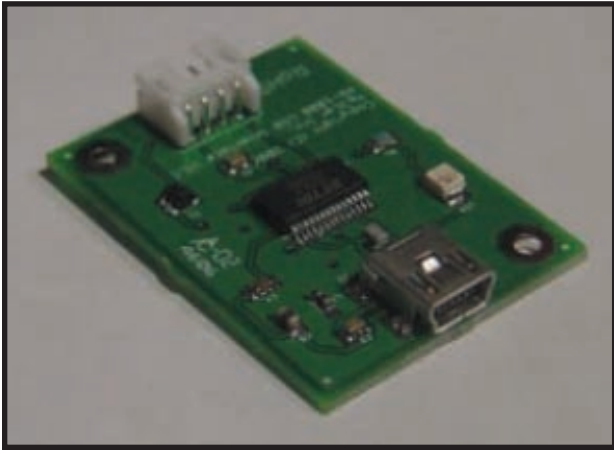


Photo 5—The finished product! In hindsight, wasn't that pretty easy?

the ability to manufacture gold contacts.

GERBER FILES

Your fabrication house will require a set of computer files that describe your board's physical layout. The most common format is the Gerber RS-274X format where you produce one file for each layer on the board. (Gerber is the company that created the standard.)

The PR-1000 board is a two-sided board described by the following files: Top copper (file extension = .top) gives the shape of the traces and pads on the top side of the board where the components are placed. (There are no standard extensions for Gerber files. Just make sure you agree with your manufacturer about them.) The top side is often called the "component layer" because most boards have only components on the top. Bottom copper (.bot) gives the traces for the back side of the board. This is often called the solder side.

The top and bottom solder masks (.tsm and .bsm) are thin solder-phobic plastic layers that cover the board everywhere except on the surfaces that are to be exposed to solder. It restricts the solder to the solder pads and enables them to be more densely packed without shorting pins. Top and bottom silkscreens (.tsk and .bsk) are the white printing you see on circuit boards that give the names and outlines of the components. The top and bottom cream layers (.tcr and .bcr) define where the solder paste is applied during assembly. This is often used to manufacture a stencil or thin

metal sheet with holes where the solder is applied to your board. Outline (.oln) defines the shape of the board. Excellon drill information and data files (.dri and .drd) describe the pattern of holes to drill in the board. (Excellon is the company that standardized these files.) The holes can accommodate component pins or connect different layers of the

board electrically (vias in this case).

For the board assembly, you will also need to include a centroid file also called an XY file. (The centroid is the center of your component.) This is an ASCII file generated by your design software with coordinates for the centroid and rotation angle for each component on your board. The file is used by robots to position the components automatically for soldering. A description of the format can be found on the Screaming Circuits web site (www.screamingcircuits.com).

After you've generated your files, you can view them in a Gerber viewer (such as GerbView or Viewmaster) to ensure that there are no mistakes. If you are on a budget, you may wish to use Viewmaster's free version, Viewmate.^[2]

Finally, you should also include an ASCII file with any specialized instructions, including special precautions, whether or not the board is RoHS, the types of materials to use, and information about how to contact you if there are questions.

PANELIZING

Your next task is to "panelize" your design. This means you have to produce a new set of Gerber files for a larger board (a panel) that contains 10 to 20 of your smaller circuit boards side by side. Photo 2 shows a panel of PR-1000 boards prior to assembly. After assembly, the panels can be separated into individual boards.

Higher-end EDA packages and CAD

programs can design a panel from your single board given the dimensions of the desired panel. In addition, most PCB fabrication houses will produce one for you from your Gerber files for a nominal fee. If you have any special dimension requirements, however, be sure to include them in your instructions to the fabrication house.

There are two basic methods of spacing the boards on a panel. With tab routing, a router cuts around the perimeter of each board, leaving a small tab of material between boards (see Photo 3). After assembly, the tabs can be cut to separate the panel into individual units. You can also drill perforations in the tabs so they can be broken by just "snapping" the board. This is called "tab with perf." The second method is called v-score. An angled bit cuts a v-shaped groove around the perimeter of each board without cutting completely through the panel. The v-score takes up less space, but it can be used only to make square boards, unlike tab routing, which can produce any shape. Pallet is a synonym for panel in the PCB industry.

ASSEMBLING YOUR BOARD

Assembly houses have two methods for procuring components: turnkey and consignment. With turnkey, you send them a list of all the parts and they buy and handle everything. Turnkey is less of a hassle. For small projects, you pay for the assembler's time to identify and purchase parts for you. You may, however, get an economy of scale for common parts. With consignment, you send them a list of all the parts and you supply them the exact parts you want. Consignment is often more useful if you like to control your own inventory and you have done a good job of specifying your parts.

I used consignment because I'm sharing parts between several boards in order to create my own economy of scale. I also like to control every detail of the process, so consignment is preferable. Some of my parts are shown in Photo 4. Standard packaging modes include: tape, reel, tube, bulk parts, and tray.

Modern electronic components are typically packaged in tape—a strip of plastic with small bubbles containing

one component each. Tape width varies with the size of the component it contains and it has a perforated edge so robots can feed the tape.

A reel is a spool of tape. Some robots require or work faster with spools rather than loose tape because spools don't get tangled.

Some parts (usually microchips) are best delivered in a tube. Tubes can also be fed into robots.

Bulk parts are loose in bags or boxes. Someone will have to place them by hand. This is acceptable for small orders.

Larger parts are sometimes distributed on trays that hold a few dozen parts in a sheet. Robots can then pluck the parts out of the trays during manufacture.

KITTING

When doing consignment, you have to "kit" the parts before sending them to the assembler. Simply put, each specific part goes in its own static-proof plastic bag with a label giving the part number included in the bag and the part count. This way, the assembly house can look down the list and say: "Now we need the part GNM12345. Where is it? Oh! Here it is!" They can open up the bag, remove the parts, and insert them into the robot.

Because the robots are using feeder systems, you will often "lose" a few inches of tape. Depending on the robot and your component dimensions, you may lose varying numbers of parts on the order of 10 to 100. Talk to your assembler about how many extras to supply. Once all of the bags are filled, box and send them to the assembler with the BOM.

TESTING & REWORKING

After your board has been assembled, the assembler will commonly test your board for electrical connectivity. It does this with your netlist and a robotic machine that touches all of the pins that should be connected on your board. If any failed connections are found, the board is marked as bad. It can also do additional tests for more complicated boards including microprocessor testing.

Reworking is the process of desoldering and resoldering specific compo-

nents on a board. This can become necessary for a number of reasons, and not just because you made a mistake and put the wrong component down. For example, modern ball grid arrays (BGAs) are very sensitive devices that have very tiny solder connections to the board. (A BGA is a type of microchip with small balls of solder on the underbelly rather than pins.) In many cases, it is not possible to attain 100% defect-free production of boards with BGAs on them. In this case, an assembly house may process all of the failed boards using an X-ray machine to determine if the BGA's solder connections are open and then use a rework station to repair it.

END RESULT

In the end, you will have a professional board. You can see a finished PR-1000 board in Photo 5. All of the source files for this project (i.e., schematics, Gerbers, notes, etc.) are posted on the *Circuit Cellar* FTP site. You can also purchase a finished board from MP3Car (<http://store.mp3car.com>). ☱

Zack Gainsforth (zack@gainsforth.com) earned a Physics degree from the University of California at Berkeley. He is currently working on analyzing cometary dust from the NASA Stardust mission and circuit board development for MP3Car. Zack's interests include electronics, space science, building things, and sushi.

PROJECT FILES

To download code and additional files, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/208.

REFERENCES

- [1] AlternateZone.com, "PCB Design Tutorial," 2004, <http://alternatezone.com/electronics/pcbdesign.htm>.
- [2] PentaLogix, "Viewmate—Free Gerber Viewer," 2007, www.pentalogix.com/Download/download.html.

SOURCE

PR-1000 Haptic knob
Immersion Corp.
www.immersion.com

AC SERVO SYSTEM

Including:

- 250 watts AC servo motor
- 14 bits absolute sensor
- 250 watts AC driver

As low as \$399.99



www.dmm-tech.com

DMM Technology Corp.

ANDRE LAMOUE'S
XGAMESTATION
DO-IT-YOURSELF VIDEO GAME SYSTEM

Inspired by the
Atari 2600, Apple II &
Commodore 64!



SX52 @
80 MIPS

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
(512) 266-2399 SUPPORT@NURVE.NET

PIC-SERVO MOTION CONTROL

MOTION CONTROLLERS FOR
BRUSH, BRUSHLESS AND
STEPPER MOTORS.

- controller chips
- controller boards

www.picservo.com

JEFFREY KERR, LLC



Motor Driving for a Robotic Arm

While working on a project to provide motor drivers for a robotic arm, Jeff needed an interface to control each motor. Although he could have used a high-level software package, he wanted something simpler just to prove the hardware. This is the story of how he got the job done.

I don't consider myself a shopper. The word "sale" doesn't make me drool. Although I admit that I have ogled at the site of a hot motorcycle, automobile, and even the occasional high-tech gadget, I consider myself rather frugal. When the need arises to go to a store, I make a point to see how little time I can spend inside. I rarely roam the aisles, except when accompanying my wife, Beverly. She needs to touch every item she sees. I find this process painfully boring and prefer to roam the store on my own if forced to remain longer than my usual two minutes. To kill time, I look for interesting deals that might include a tool (e.g., a cheap high-torque screwdriver for its motor), some electronics (e.g., an answering machine for its digital recorder), or a toy (e.g., a game with an unknown position sensor) that could be educational. Anything that enters my home must first undergo an interrogation with a screwdriver.

Recently, while working on a project to provide motor drivers for a robotic arm, I realized that I needed an interface that would control each motor and provide position feedback. Although this would eventually come from some high-level software that would calculate trajectories for each joint and enable the arm to get from here to there, I needed something simple just to test the hardware.

The controller I chose can run Linux with a USB interface for the peripherals. My driver design implemented a USB interface. This enables me to access my

hardware through my PC. I used Liberty BASIC to program an application to give me the ability to move up to eight motors and retrieve feedback from each (see Photo 1). This column is not about that application. It works great for its intended testing purpose. This is about using one of the deals I found. Actually, I didn't find the deal in a local store. It came from one of the many e-mail flyers I receive from distributors. The item is a six-channel RC transmitter/receiver package that costs \$30.

Innovation First, the official supplier of control systems for the FIRST Robotics Competition, designed the unit for the competition. It is still sold new for around \$100 (www.vexlabs.com) as part of the Vex Robotics Design System. You may have seen the Vex Robotics system at RadioShack stores, although RadioShack has discontinued the product line.

OWNER'S MANUAL

My purchase arrived within a few

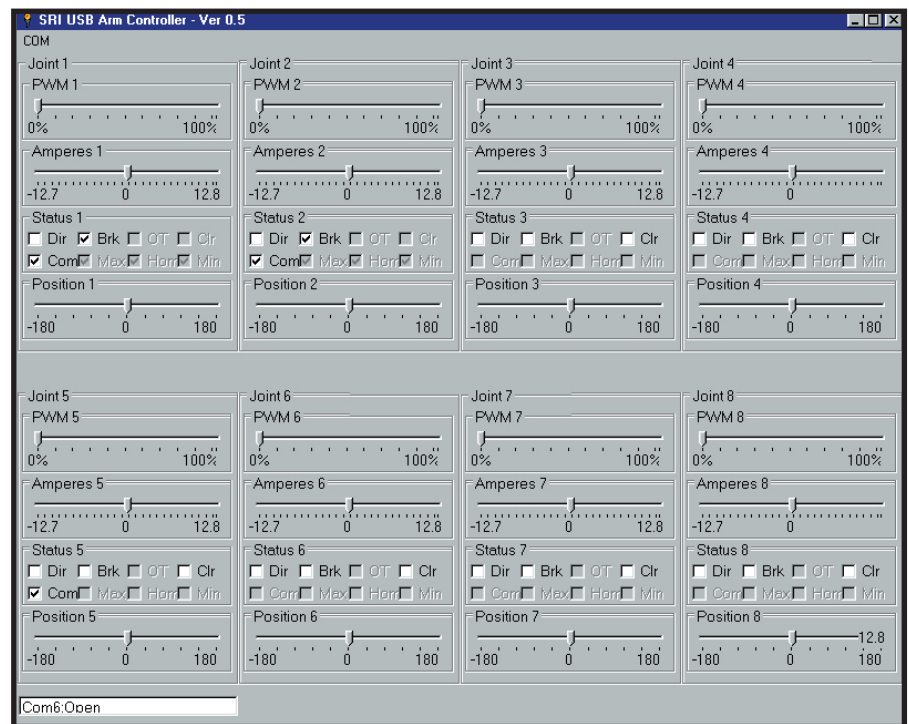


Photo 1—This PC application was designed to control eight motors. The top slide potentiometer sets the PWM value. The second potentiometer reports motor current. Checkboxes enable the control of direction and report the status of the brake and limit switches. The encoder position for each motor is also displayed using a slider.

days. Inspecting it didn't take long. The transmitter, about the size of the most recent *Harry Potter* book, required eight AA cells. The receiver, the size of a bag of Bertie Bott's Every Flavor Beans, had no battery compartment. In fact, the only connector on either unit was a four-pole RJ-9. The single sheet of paper included wasn't much more than the required "FCC wants you to know" fine print. That's right: no documentation whatsoever. A quick note off to Vex brought back the reply, "proprietary circuit." Those guys weren't about to help.

It looked like the transmitter and receiver connectors might be the same, so I started with the transmitter. At least it had a place for batteries and I could probe the connector for sig-

nals. My screwdriver revealed the interior PCB with labels at the connector:

5, signal, GND, and tether detect. (I recall that many RC flying instructors operate with a tethered controller for the trainee. This enables them to take control of the aircraft if the Padawan puts the aircraft in jeopardy.)

The output was an open collector requiring a pull-up. The output is shown in Photo 2. It would make sense to have the transmitter and receiver connectors be compatible, but I proceeded with caution anyway. By limiting the supply current, I tried to prevent a disaster. As it turns out, the receiver uses a similar pinout.

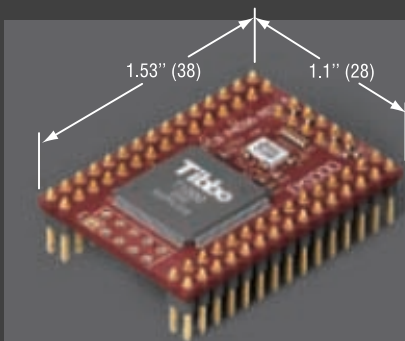
This project incorporates the wireless RC unit to control the motor driver as an alternative to the originally designed USB wired interface. Although the wired inter-

Register #	Function	Direction	Data
0	PWM (MSB)	Write	0-1023 (0x000-0x3FF)
1	PWM (LSB)		
2	Control	Write	Bit 0 - Direction (1 = Fwd, 0 = Rev)
			Bit 1 - Brake (1 = On, 0 = Off)
			Bit 2 - n/a
			Bit 3 - Clear encoder (1 = to clear)
			Bit 4 - n/a
			Bit 5 - n/a
			Bit 6 - n/a
	Bit 7 - n/a		
3	Current	Read	0-2.55 Amps (0x00-0xFF)
4	Status		
		Bit 1 - Brake (1 = On, 0 = Off)	
		Bit 2 - Over temp (1 = Yes, 0 = No)	
		Bit 3 - Encoder zero (1 = Yes, 0 = No)	
		Bit 4 - Comm (1 = Yes, 0 = No)	
		Bit 5 - Maximum limit (1 = Yes, 0 = No)	
		Bit 6 - Home (1 = Yes, 0 = No)	
	Bit 7 - Minimum limit (1 = Yes, 0 = No)		
5	Odometer (MSB)	Read	0-32565 (0x0000-0x7FFF)
6	Odometer (LSB)		

Table 1—Seven bytes of data are reserved for each of the eight motors. The first registers set speed (via PWM value) and direction (via control). Additional controls are for the motor's brake and clearing of the odometer (an optical encoder counter). Various status bits are reported from each motor driver as well as the motor's position (via an optical encoder counter).

Tibbo
TECHNOLOGY

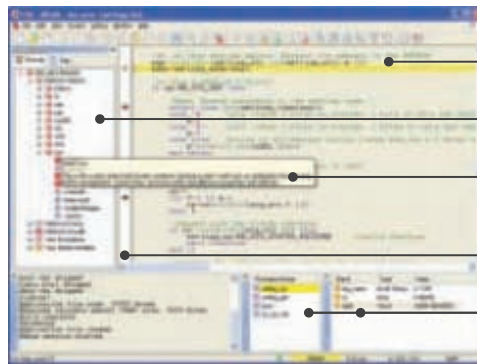
Build your next automation project around our EM1000 BASIC-programmable Embedded Module



- 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)

- 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)

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



- Write in familiar BASIC language
- Inspect objects, procedures, and variables
- Code faster with auto-completion and code hints
- Set breakpoints, execute step-by-step, etc.
- Monitor the state of variables and stack

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

face is fine for a robotic arm that isn't wandering far from its controller, if the motor driver were to be used to drive the wheels of a robot platform, then wires would be frowned on. Have you ever seen *Robot Wars*?

RC TIMING

I've done a few projects involving RC servo timing, so I won't go to great lengths describing it. However, here are a few of the high points. The basic RC servo is a motorized arm that has a limited rotation (normally 90° or 180°). A potentiometer rotates with the arm to provide position feedback to the servo. The servo must see a positive 1- to 2-ms pulse every 20 ms (or so) to remain active. A 1-ms pulse equates to a minimum rotational position (0°), while a 2-ms pulse stands for the maximum rotational position

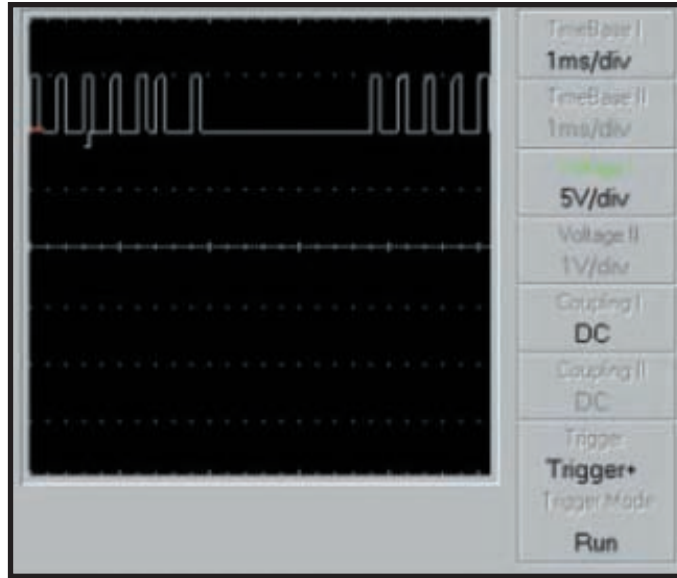


Photo 2—The Vex transmitter/receiver output waveform consists of a high fixed pulse width of 500 μ s followed by a low variable pulse width between 500 and 1,500 μ s for each of the six channels. The seventh positive/negative pulse puts about 10 ms of silence between channel reports. The first four channels are reporting idle (1,000 μ s), the fifth channel is showing 500 μ s (minimum timing), and the sixth channel is showing 1,500 μ s (maximum timing).

(90° or 180°), with any timing between 1 and 2 ms being proportional between the maximum and minimum positions.

While we are not actually going to be driving RC servos in this application (the motor driver board requires data as inputs and outputs PWM control), knowing the timing involved is critical to translating that into data. As it turns out, the datastream sent by the Vex transmitter and output by the receiver is a combination waveform of the six RC channels in sequence (see Photo 2). While the format is inverted, the pulse timing is correct. In the transmission, a channel begins with a positive pulse (500 μ s) with the actual timing pulse (1 to 2 ms) beginning on the falling edge of that pulse. The timing for each successive channel is

packed together, one after the other, with a seventh positive pulse indicating the end of the timing of the

PCB-POOL®

SERVICING YOUR COMPLETE PROTOTYPE NEEDS

Price Example: 16 Sq-Inches (double sided pth)

2 Days: \$ 90.00

8 Days: \$ 22.50

Standard PCB-Pool Service
SIMPLY SEND YOUR FILES AND ORDER ONLINE!

New Service:

WATCH "ur" PCB®

Save vital time on design errors in advance of receiving your Prototype. View high resolution photographic images of your PCB during each production stage. Be one step ahead, use our realtime PCB monitoring service.

WWW.PCB-POOL.COM

Download our free PCB software
www.free-pcb-software.com

ROHS / WEEE conform

LEAD FREE

Tollfree USA : 1877 390 8541

sales@beta-layout.com

TARGET
PROTEUS
Protel
Electronics
OrCAD
Spring
Easy-PC
PTAS
CircuitCade

sixth channel. The series is repeated about every 20 ms.

This project adds an alternate input to the motor driver board, which comes from the Vex receiver. When selected, the alternate input measures the timing of each channel's output pulse widths and translates those times into data the driver board can use for each motor.

USB INTERFACE

Presently, the motor driver board uses a USB interface to handle data to and from a host application. For testing purposes, it has been an application running on a PC. The design enables control of up to eight motors driven from data in a table located within a microcontroller with a USB peripheral (see Table 1). The data is updated to each motor driver using an SPI. The data is replicated for each of the eight motor drivers.

The design of the motor driver board has two flavors. The first has four (less than 3 A) motor drivers and the second has two (greater than 3 A)



Photo 3—The Vision-Arm's shoulder (Smart Robotics) consists of rotation and pivot joints. Machined aluminum parts keep weight down to a minimum.

motor drivers. They can be stacked for combinations of driving motors of various sizes (up to eight). The USB interface uses a 2-byte format to send and receive data to the tables. With this format, you can write/read to and

from either the data (byte) pointer or the data (byte) itself. The table's data is self-refreshing (i.e., it is constantly being sent to and updated from each of the motor drivers). So all you need to do for this project is keep the data tables updated with the latest information provided by the RC servo datastream. There are 3 bits that you need to handle beyond the direction bit (and PWM). You must ensure that the brake bit is handled properly (based on the PWM value). The maximum and minimum limits are monitored to prevent moving beyond these mechanical stops (if implemented for any particular motor).

SERVO TIMING TRANSLATIONS

The transmitter uses spring-loaded joysticks for the first four channels. This means that the idle position of each channel is centered. The timing for all idle channels will be halfway between 1 and 2 ms, or 1.5 ms. This will be considered off (no forward or reverse power). From 1.5 ms up to 2 ms will be interpreted as "stopped to full

Self-powered Radio Modules

flexible / battery-free RF links to connect your Wireless Sensor Networks

Self-powered WIRELESS™	Energy Harvesting Solutions for sensors & control applications
Reliable Range - 150' indoors - 315 or 868 MHz - Solar operates for days without light - Easy-to-use	Solar-powered for Sensors • temperature • light levels • pressure • water • humidity • vibration • current • occupancy
	Mechanical-powered for Controls • on/off control • light switches

Wireless Sensor Development Kits
Loaded! new low price

AD HOC ELECTRONICS™

AdHoc101.com

See how far Self-powered Wireless can go! (801) 225-2226

@LINX Engineering = Fun

Looking for a special place where your skills can be applied in a non-bureaucratic, visionary environment?

Located in beautiful Southern Oregon, Linx is the perfect place to combine a love of engineering with a high quality of life. For more than a decade, Linx products and services have enabled over ten thousand customers to easily and cost-effectively add wireless capabilities to their products. Linx is expanding, and so is our vision for simplifying complexity. If you are looking for employment in a close-knit team where your skills will be utilized, your ideas valued, and your efforts appreciated, we would love to hear from you.

Now Hiring!

Senior RF Engineer
RF Engineer
Application/Sales Engineer
Test/QA Engineer
Software/Firmware Programmers
Device Networking Specialist

Join the Fun opportunity@linxtechnologies.com

LINX TECHNOLOGIES

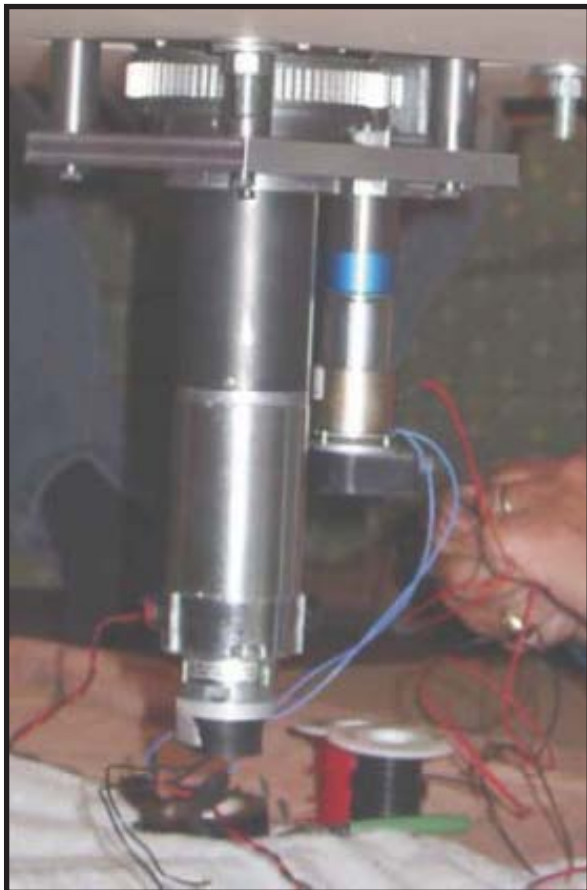


Photo 4—The movement modules for the shoulder joints comprise (from top to bottom) a gear box, a DC motor, a brake, and an optical encoder. The brake, which must be powered to be released, prevents the joint from moving while the DC motor is powered off. This conserves power when the arm is inactive.

forward.” Note that from 1.5 ms down to 1 ms will be translated as “stopped to full reverse.” The relationship between the Vex timing and the timer count (and ultimately PWM and direction) is shown in Table 2.

I like to use a weighted average to reduce any jitter that may be in the transmitted timing signal or in my sampling routine. In this case, I

used a 1/8 average:

$$\text{New average} = \frac{(7 \times \text{old average}) + \text{new count}}{8} \quad [1]$$

To keep things simple, I broke the timing calculation into two possibilities. For counts 4,000 and up (forward), I use:

$$\text{PWM value} = \frac{(\text{count} - 4,000) \times 1,023}{2,000} \quad [2]$$

For counts less than 4,000 (reverse), I use:

$$\text{PWM value} = \frac{(4,000 - \text{count}) \times 1,023}{2,000} \quad [3]$$

Because the 16-bit timer’s tic is 250 ns, I know the timer will overflow at about 16 ms (i.e., $65,536 \times 250$ ns). If this happens, I know the Vex receiver is no longer outputting data. This might be used to place the system in Safe mode. Because the maximum RC timing is approximately 1,500 μ s (6,000 tics), anything over that (with a safe margin added in) will indicate the end of the cycle (the last channel is finished). At this point, the channel count can be cleared in

Got Serial, Need Network?

Bluetooth
Qty 1
\$175

Ethernet
Qty 1
\$99

Wireless
Qty 1
\$199

Volume Discounts Available



gridconnect™
www.gridconnect.com
+1 800 975-4743

ZigBee™ Certified Wireless Modules



XBee™ and XBee-PRO™



ZigBee™ Mesh Networks

- Range up to one mile
- Worldwide approved 2.4 GHz
- Modules start at \$19

Order your development kit online today!
from \$129



Toll-free 866-765-9885
www.zigbeemodule.com/cc

Pulse time	Tics	PWM value	Control direction	Control brake
0.5 ms (minimum)	2,000	1,023	1	0
0.6 ms	2,400	823	1	0
0.7 ms	2,800	623	1	0
0.8 ms	3,200	423	1	0
0.9 ms	3,600	223	1	0
1 ms (idle)	4,000	0	0/1	1
1.1 ms	4,400	223	0	0
1.2 ms	4,800	423	0	0
1.3 ms	5,200	623	0	0
1.4 ms	5,600	823	0	0
1.5 ms (maximum)	6,000	1,023	0	0

Table 2—The Vex transmitter/receiver signals have a minimum low pulse width of 500 μ s and a maximum low pulse width of 1,500 μ s. The pulse widths are measured by a counter with a 250-ns time base and translated into PWM values as well as a direction bit. Braking is applied (or power removed) when the PWM is at a value of zero.

anticipation of the next round of timings.

It is important to be aware of the execution times for various calculation routines. For the above calculations, you need to use a 16- \times 16-bit multiply and a 24- \times 16-bit divide. Typical execution times for these would be 256 and 539 cycles, respectively. With a 500- μ s pulse between timings and a 250-ns execution clock, that gives only 2,000 instruction times to do the calculations. You don't want to squeeze in more code than you have time to execute. There are some shortcuts you can use. For instance, to do the averaging, it takes less time to do seven adds, plus one add (for the multiply), and shift right 3 bits (for the divide), than it does to use stock multiply and divide routines. Precious time is saved by shifting left to multiply (by factors of two) and right to divide (by factors of

two). I am using a Microchip Technology PIC18F2320 microcontroller that has an 8 \times 8 hardware multiply instruction and can save cycles. But it's always a good idea to look at the big picture to make sure you are not asking for more than your microcontroller can give.


The Vex transmitter/receiver has a pair of buttons on the back for the last two channels. Using the buttons is similar to pushing a joystick axis to maximum and minimum. The idle timing (1 ms) for the channels moves to 0.5 ms (minimum time) with one button and 1.5 ms (maximum time) with the other button. Because I can control up to eight motors and the Vex offers four analog plus two digital (if you will) timings, I use one of the last two channels to redirect the four

analog channels to either the first four or the last four motors. When a redirection occurs, all of the motors are set to off (zero/idle). This way, no motor is left running without a source of control.

FINALE

Preliminary tests of the shoulder and upper-arm joints were a great success (see Photos 3 and 4). Although I set up the test with the PC application using the USB interface, it quickly became apparent that using the mouse to adjust slide potentiometers on the application screen became tricky once more than one axis was moving at the same time. It was easy to forget which potentiometer controlled which motor, making it difficult to react quickly to emergency situations. This was a lot like trying to control a bull in a china shop.

As luck would have it, I had the Vex transmitter and receiver with me. They were mounted on a wheelbase to show off the motor driver PCB being used in a mobile application (see Photo 5). With a few swaps, I was able to use the Vex transmitter to control the arm joints. This turned out to be a superior way of "demo-ing" arm motor control.

I've learned to keep an open mind. You never know when something will turn out to be really useful in a way you'd never intended. And so much better when it turns out to be inexpensive as well. 

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 through the magazine (jeff.bachiochi@circuitcellar.com) or his web site (www.imaginethatnow.com).

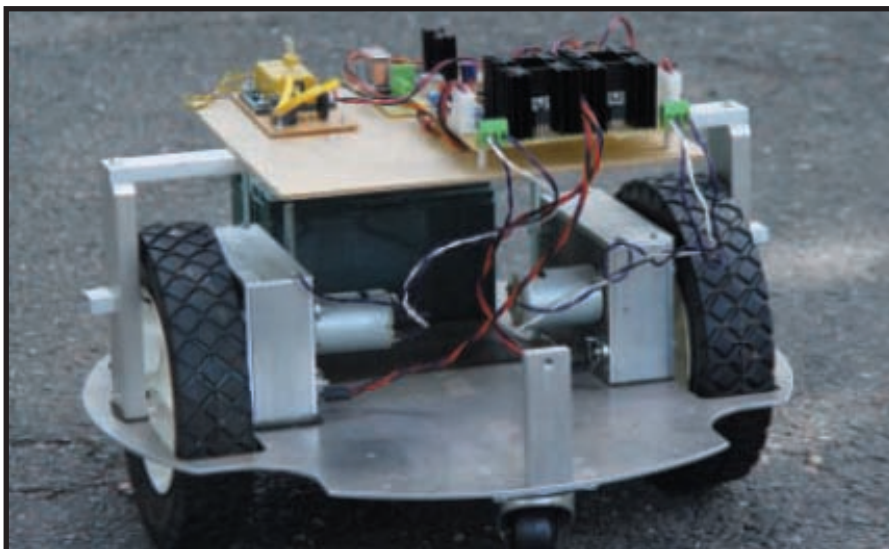


Photo 5—This mobile robot platform's DC motors (seen on the insides of each wheel's gear box) are driven by the same motor driver PCB designed as the arm controller. It is temporarily mounted to the top of the platform to the right of the Vex receiver.

SOURCES

PIC18F2320 Microcontroller
Microchip Technology, Inc.
www.microchip.com

Vision-Arm
Smart Robots, Inc.
www.smartrobots.com

Transmitter and receiver add-on kit
Vex Robotics
www.vexlabs.com



A Flash in the Pan

George explains how to use C language when interfacing flash memory devices in an embedded system. He describes how to write C code to detect, read, write, and erase flash memory.

According to Phrasefinder, a “flash in the pan” means “something which disappoints by failing to deliver anything of value, despite a showy beginning.”^[1] Let’s see how we do with this month’s work.

Before we jump into this column, let me share some reader e-mails. Several offered suggestions on how to improve my presentation. Their e-mail addresses (someone@vmware.com and another@freescale.com) told me that these folks are black belt C types. Others (someone@someschool.edu) asked for advice on an evaluation board for C programming. That’s the beauty of C: users at all levels can participate. One applications engineer from a major electronics distributor told me that he liked the articles but he could only read C, not write C. Hmm...

This month, I will explain how to use the C language to interface flash memory devices in an embedded system. Flash memories are very popular in embedded systems. The Texas Instruments MSP430 that we started with has flash memory built in. The NetBurner module we worked with last time has its flash memories off the processor. Both systems can save program and data space in the memory. The tools are so easy to use that you do not even need to know how to program the flash memory to save and update program contents. Utilities are also available to support the flash memory devices in both systems.

Let’s say you need to include flash memories in your next design, saving program, user messages, or data. What then? Well, this month I’ll describe how to write C code to detect, read, write, and erase flash memories from Intel. I want to cover this topic in some detail, so I’ll make this a two-part presentation.

In the next installment, I will cover Spansion and how to integrate both devices into the design. Because I want to present tested code, and the unexpected delays that come when developing new products, the Spansion portion of this discussion will have to wait until issue 212.

SYSTEM REQUIREMENTS

Let’s lay out our system and list some requirements. The code I’m going to include in this article is using an Altera NIOS II processor. The processor is embedded in an FPGA as intellectual property (IP). The application is a large messaging system and the FPGA provided lots of support for all of the custom hardware interfacing required. But, for our purposes, it’s just a 32-bit CPU with a 16-bit external data bus and a 32-bit external memory bus.

The flash memory requirements include: identifying the device installed, erasing the entire device, erasing one particular sector, and programming data into the device.

The two types of flash memory we are going to work with have significantly different specifications. I urge you to look at them. Intel can be found at www.intel.com/design/flcomp/datashts/306666.htm and Spansion at www.spansion.com/datasheets/s29ns-n_00_a13_e.pdf. The Intel devices are for the J3 family. Intel has older and

newer families. Most, if not all, should use the same interface that I’m describing here. I’ll describe the Spansion interfacing in more detail in the second part of this discussion.

The Intel device can be 32, 64, or 128 Mb, while the Spansion can be 64, 128, or 256 Mb. Any member of the family can be soldered to the board. I also believe Spansion is planning to go larger in the same package. These are TSOP (56 pin) packages. We were trying not to use the BGA packaging for ease of manufacturing. We don’t have the X-ray equipment necessary for BGA inspection.

Both devices are NOR-type flash memories. The type of memory found in MP3 players is NAND. NOR has individual memory access and can be used to hold a program, while NAND has more serial access and typically uses a memory controller. NAND is less expensive per bit, but both prices are racing downward rapidly. Refer to http://en.wikipedia.org/wiki/Flash_memory for a more detailed description of the types.

Note, for clarity I’m going to write the 32-bit address in hexadecimal notation with a space to make it more readable. So 0x0000 0000 is location 0x00000000. And 0x0800 0000 is location 0x8000000. Also, in my memory-address notation, the LSB will reference the byte. The hardware design is using only word access and you should see only even addresses while you’re debugging.

The flash memory devices have the capability of reading and writing each individual location (one word at a time or in blocks for speed). The writing process changes bits from ones to zeros. Once a bit is changed to a zero, it cannot be written back to a one. In order to make that change, an entire sector

	Intel	Spansion
1	0x0800 0000	0x0800 0000
2	0x0A00 0000	0x0C00 0000
3	0x0C00 0000	0x2000 0000
4	0x0E00 0000	0x2400 0000

Table 1—Some flash memory device system addresses.

Listing 1—This is a list of the prototypes for the procedures found in the flash low-level module. You'll note routines for Intel, Spansion, and one for both that sorts out which device is installed. You'll probably have similar procedures in your design.

```

INT16 DetectFLASHDevice_INTEL(UINT16 *BaseAddr, INT16 WhichDev);
INT16 DetectFLASHDevice_SPA(UINT16 *BaseAddr, INT16 WhichDev);
INT16 DetectFLASHDevice(UINT16 *BaseAddr, INT16 WhichDev);
void DetectALLFLASHDevices(void);

INT16 EraseFLASHBlock_INTEL(UINT16 *FLBaseAddr, INT16 WhichBlock);
INT16 EraseFLASHBlock_SPA(UINT16 *FLBaseAddr, INT16 WhichBlock);
INT16 EraseFLASHBlock(UINT16 *FLBaseAddr, INT16 WhichBlock);

INT16 WriteFLASHWord_INTEL(UINT16 *WrAddr, UINT16 Data);
INT16 WriteFLASHWord_SPA(UINT16 *WrAddr, UINT16 Data);
INT16 WriteFLASHWord (UINT16 *WrAddr, UINT16 Data);

INT16 WriteFLASHBuf_INTEL(UINT16 *WrAddr, UINT16 *dataArray, UINT16 n);
INT16 WriteFLASHBuf_SPA(UINT16 *WrAddr, UINT16 *dataArray, UINT16 n);
INT16 WriteFLASHBuf(UINT16 *WrAddr, UINT16 *dataArray, UINT16 n);

```

needs to be erased. The sector size for Intel is 1 Mb. And a 128-Mb device has 128 evenly sized sectors.

INTEL FLASH

Let's start with the Intel 128-Mb device. The first device in our design is located at 0x0800 0000, and the last is at 0x0E00 0000 (see Table 1). It contains up to 128 sectors (also called blocks in the code) and each is a uniform size of 0x20000 bytes. The device can be accessed in either byte (8-bit) or word (16-bit) mode. I'll only be using word (16-bit) for this design.

Writing or programming the Intel flash memory is accomplished by writing a series of commands to specific memory locations. You will see this as we get into the design. Monitoring the progress of the operations is accomplished by reading the memory at specific locations. During an operation, status information will be presented instead of the memory contents.

The operations we are going to perform are Reset, Write, and Read ID. The Read ID reads the flash memory device manufacturer's code and device ID. The operation identifies which device from which manufacture is installed in the system.

The hardware design has four flash memory positions. I've designed the NIOS CPU to have four memory chip select lines. One line for each of the flash

memory devices. The memory can either be populated with Intel or Spansion devices. The board must contain all of one type, but can have different members of the same family on a board. Because memories are not pin compatible, how can we have this diversity? Well, we started production with Intel and shipped hundreds of units. We then made a change to the design and switched to Spansion. But, both versions of the hardware need to be supported. And I bet as time goes on, it's possible that we could even include another manufacturer. The four flash memory devices are at the system addresses listed in Table 1.

SOFTWARE LEVELS

I divided the software into several levels. LowLevel, MidLevel, TopLevel, and Reports. The LowLevel routines work with the detailed addresses and bits from the datasheets. Refer to Listing 1 for those routines.

The MidLevel routines convert the details of the specific design into the LowLevel routine calls. The original design is a messaging system. Messages are located in flash memory. The MidLevel procedures translate the message numbers to flash memory locations, keep track of deleted message space, and when a complete sector is no longer used and can be erased. It makes no sense to include

the MidLevel routines in the discussion.

The TopLevel routines contain the routines that the user and other programs would interface with (see Listing 2).

First, let's look at the code to DetectALLFLASHDevices(). It makes a call to void InitFLASHTables(void). In that routine, I clear out all information about all flash memory devices detected. I'm clearing out two data elements. The first is an array:

```

INT16 FLASHStatus[MAX_BASE_FLASH_DEV];
// 0,32,64,128,256 Mbits

```

The array holds a number representing the number of megabits the flash memory device supports. When we move to Spansion, the number can also be 512 and even 1,024 (yes, a 1-Gb device). Now we're talking. The constant MAX_BASE_FLASH_DEV is defined as four.

I also initialize the structure:

```

struct FLASH_DEV FlashInfo[MAX_BASE_FLASH_DEV];

```

This structure is defined in Listing 3.

I can record a string for the name, the manufacturer's number, the device ID number, the size in megabits, and a variable about block lock status. I do not use the block lock feature in either of the devices. If you've got a keen eye (all you C black belts), you might have noticed that FLASHStatus[] and FlashInfo[]. Size are redundant. They hold the same information. This is true. I never went back to clean up my design.

You should recognize each of the components of the C code that I have described. If not, I suggest you go back to the beginning of this series.

Next, the routine DetectALLFLASHDevices() calls:

```

INT16 DetectFLASHDevice(UINT16 *BaseAddr, INT16 WhichDev);

```

The routine performs the functions shown in Listing 4.

DetectALLFLASHDevices knows that there are four devices and provides the base address of the potential device and the index into the information table to place the results. Now

```

INT16 DetectFLASHDevice(UINT16 *BaseAddr, INT16 WhichDev);

```

Listing 2—This is a list of the prototypes for the procedures found in the flash top-level module. These provide an interface between the functionality of the product and the detailed operation on the flash memory devices.

```

void FLASHStartup(void);
void ReportFlashDevices(struct COM_DETAILS *cp);
void ProgramDownload(struct COM_DETAILS *cp); // Download a program
void MsgDownload(struct COM_DETAILS *cp); // Download a user message
void ReportFlashDevices(struct COM_DETAILS *cp);

```

knows that there can be an Intel device, a Spansion device, or an empty position. It first tries to detect an Intel device. If the size returned is 0, it means that either nothing is soldered into that position or a Spansion device is soldered in. The Spansion device would not respond to the detect Intel command set. If the size returned is 0, then we will try to detect the Spansion type of device. And if there were other possible devices, the code to detect them would go here.

Now let's look into the DetectFLASHDevice_INTEL(BaseAddr, WhichDev); routine (see Listing 5).

BaseAddr is defined as a pointer to an unsigned 16-bit entity. And FLASH_CMD_READ_ID is the constant 0x90. This is straight from the Intel manual. A write 0x90 to flash memory location 0000 0000 is the Read Identifier command. I save a copy of the BaseAddr. Next, I read the manufacturer's ID from the flash memory device. Wait a minute. I can hear you say, "It's a memory. Why doesn't it just return the contents of the memory at that location? How does it know to return the manufacturer's ID?" Good point. The writing of the command to the BaseAddr caused the device to no longer be in the "I'm a Memory" mode, but instead "I'm a Port Into the Memory Controller State Machine" mode. If you were trying to execute code (or fetch data) out of this memory at the same time you were trying to identify it, the reading of the code (or data) would fail. That's why I commented out the code to disable interrupts. As it turned out, in this design, I'm executing the flash routines out of RAM. The code was transferred from flash to RAM at boot. I never have to worry about executing from flash at the same time I'm programming flash.

If the mfr variable is 0x89 (Intel), then I fill in the variables with the specific Intel information. I also check to see if the block is locked. I never came across a locked block, so I don't know if this code would work. The last thing we do before exiting the routine is:

```
ResetFLASH_INTEL(BaseAddr);
// Reset the FLASH Device
```

This writes the command to reset the flash memory device and puts it back into the "I'm a Memory" mode.

Listing 3—This is the structure for holding all of the information about the flash memory device installed in the unit. Remember that a structure is a collection of different types of variables. And this structure is an array for the four possible locations for flash memory devices.

```
struct FLASH_DEV FlashInfo[MAX_BASE_FLASH_DEV]; This structure is
defines as:
    struct FLASH_DEV {
        INT8  Name[10];      // String for the name
        INT16 Mfg;          // Mfg Number
        INT16 Dev;          // Device ID
        INT16 Size;         // Size in MBits
        INT16 BlockLock;    // Is Block Lock Used
```

And just like the read ID command, it is right from the Intel datasheet.

The operation is repeated for each of the four possible flash memory locations.

NOT SO FAST

So, we've passed variables that were addresses, used those variables as pointers, wrote commands to the addresses, detected devices, filled in information tables (arrays and structures), and returned the flash memory devices to their normal operating modes. Piece of cake, right?

Let me mention here that the Intel routines have been tested in the real world, but the Spansion routines have not been debugged, so proceed with caution. I planned to get prototypes using the Spansion devices in time to debug and test the code. The untested Spansion code is posted on the *Circuit Cellar* FTP site. Use with caution. I will postpone the second part of this design until issue 212.

ERASE A BLOCK

Now let's look at what is required to erase a block of flash memory. The following routine is passed a base address and a block number:

```
INT16 EraseFLASHBlock_INTEL(UINT16
*FLBaseAddr, INT16 WhichBlock) {
```

The base address is the system address of the first location in the flash memory device (see Table 1). First, I turn on an LED indicator so that I can observe the unit's operation without resorting to

breakpoints or debug outputs. Next, I convert the block number into an offset in the flash memory device. Notice that we are working with large numbers, so I convert all the variables to INT32 when I calculate the offset. The Intel block-erase procedure writes two commands (0x20 then 0xD0) to the first address of the block that are to be erased. It then reads the data out of the flash memory at the same address and looks for BIT7 to go high. When this is complete, reset the flash so it's once again a memory device and the block is erased. All the bits in the block are erased (changed to ones). Check out the Intel documentation that I referenced in the beginning of the article. The erase operation can take anywhere from 1 to 4 s, per the datasheet. This is an eternity for some embedded systems. Intel provides a means to suspend the erase operation, making the device a memory again, and then returning to complete the suspended erase operation. Fortunately, I did not have this problem in my system requirements. Also notice that I have a timer running for the erase operation. I can't let the system hang on the erase operation. So, I wait up to 5 s maximum. The constant TIMER1_5SEC_at10MS has a value equal to 5 s given an interrupt every 10 ms. So, I expect the constant to be equal to 500.

MORE FUNCTIONS

The next function to look at is:

```
INT16 WriteFLASHWord_INTEL(UINT16
```

Listing 4—This is the code that detects flash memory devices. First, we try to detect Intel devices. If the size is returned as 0, then we try to detect Spansion.

```
Size = DetectFLASHDevice_INTEL(BaseAddr, WhichDev);
if (Size == 0) {
    Size = DetectFLASHDevice_SPAN(BaseAddr, WhichDev);
}
```


Listing 5—These are the first few lines of code in the DetectFLASHDevice_INTEL procedure. The READ_ID command is written to the base address of the device. Then, the contents of that base address are read. It will contain the manufacturer's ID information. The read operation also increments the TempAddr pointer to the next location in preparation if an Intel ID is found.

```
*BaseAddr = (UINT16)FLASH_CMD_READ_ID;
TempAddr = BaseAddr;
mfr = *TempAddr++; // Get a copy
if (mfr == 0x89) { // Mfg Code
// Intel Strata FLASH
```

```
*WrAddr, UINT16 Data);
```

This writes one 16-bit word into one memory location. The routine passes the address to write and the data. The routine starts with the LED again and then converts the write address into the base address for the flash memory device. The base address is needed as a location to issue the write command. Again, right from the Intel datasheet, the value 0x10 is written to the base address, then the actual data is written to the actual address. We then read the flash memory status much in the manner we did with the block-erase command, with the same timer protection against hanging. Write time is 40 to 175 μ s, so much quicker than block erase. And that's why I just use 2 as a timer limit. After resetting the timer, I might have just had an interrupt, so I need to look for the second one to guarantee at least 10 ms.

The Intel flash memory device supports byte operations. I've tied that signal in the PCB to be permanently Word (16-bit) access. It's not changeable and much simpler that way. So, if I wanted to do a byte write, I would need to read the existing word, mask off the bits I wanted to alter, add in my data, and do a word write. You should be able to design that routine, create a byte write routine, read the data, and do the operations required to insert your new byte. Then call the word-write operation. Another piece of cake?

This month's last function is:

```
INT16 WriteFLASHBuf_INTEL(UINT16
*WrAddr, UINT16 *DataArray, UINT16 n);
```

The Intel device supports writing a buffer of data. This operation is much faster than writing individual words. There are some rules like the data must start on an even 16-word boundary. And the specific commands are similar to what we've been performing. We

calculate a base address, write a write buffer start command, and then wait for that command to be acknowledged. If that happens, we write the data and issue a write-buffer-confirm command. As before, we wait for the success of that operation. Block write time is 128 to 654 μ s. A block write of up to 16 words is only about three to four times slower than a single word write. Cool!!

LOOKING AHEAD

Well, we've covered a lot. The Intel datasheet has flowcharts for these operations and others that I haven't used. It also refers to the common flash interface (CFI), a set of routines promising device-independent operation. ☑

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

PROJECT FILES

To download code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2007/208.

REFERENCE

- [1] The Phrasefinder, "Flash in the Pan," www.phrases.org.uk/meanings/138450.html.

SOURCE

NIOS II processor
Altera Corp.
www.altera.com



Thanks for the MEMS

Tom is always game for examining new sensor technology. This month, he introduces the MEMS-based Analog Devices ADIS16350, which he thinks has the potential to usher in a new class of applications.

In my last column, I covered some of the interesting products that I came across at this year's Sensors Expo. This time I want to take a closer look at a particularly interesting gadget that caught my eye.

As I said last month, every application starts with the ability to sense something. A new sensor can cut the cost of existing applications, thereby boosting the volume, which in turn leads to further cost reductions, increasing volume yet again. It's a cycle of success that's been a hallmark of the microelectronics revolution.

Better yet, as the cost of a sensor technology drops, doors are opened for brand new applications. Design ideas that were once "pie in the sky" all of a sudden become practical as the cost of the technology involved falls below a market-enabling threshold.

For instance, much of the excite-

ment over the Nintendo Wii video game console is attributed to its motion-sensing handheld remote that brings a new dash of realism to video gaming. This is an application that simply wouldn't exist were it not for the emergence of cost-reduced micro-electromechanical system (MEMS) accelerometers.

Now let's take a look at another MEMS-based device that has the potential to enable a new class of applications. It is hard to predict exactly what those will be, but I have no doubt that creative designers will come up with some neat ideas.

"I'VE JUST LOST MY JOB"

Just over 50 years ago, a B-29 lifted off from an airport in Massachusetts headed for Los Angeles, where it indeed arrived some 13 hours later. No biggie, except for the fact that all but the final approach was handled without the benefit of a pilot, at least not a human one. Indeed, the pilot that was along for the ride is said to have lamented, "I've just lost my job."^[1]

That day in February 1953 indeed marked a milestone in man's quest for exploration, one that would ultimately take us to the moon and beyond. The MIT scientists on board had reason to celebrate the success of their "navigator in a closet."

The concept behind an inertial navigation system is deceptively simple. All you need are three accelerometers (X, Y, Z), three gyros (X, Y, Z), and a so-called "six degrees of freedom" configuration to completely determine



Photo 1—The golden age of mechanical inertial navigation systems is represented by the Advanced Inertial Reference Sphere. That's "golden age" as in "price is no object." It's reported that AIRS contained 19,000 parts and each accelerometer alone cost \$300,000 and required six months to manufacture.^[2]

the velocity, position, and attitude of a flying object.

Simple in concept, but difficult in practice. For many years, the state-of-the-art in navigation systems relied on profoundly complicated (i.e., gim-balled) lash-ups of mechanical gyros. That first "navigator in a closet" weighed in at nearly 3,000 lb!

Thanks to the "Cold War" and the "Space Race," the development of inertial navigation systems proceeded at a fast "price is no object" pace, culminating in circa-'80s designs like the "Advanced Inertial Reference Sphere," or AIRS (see Photo 1). Designed for ballistic missiles, the AIRS could hit a target with 100-m accuracy half a world away.

As with pretty much everything else, during the '80s and '90s, the elec-



Photo 2—This is an Analog Devices ADIS16350 evaluation board. In 1953, there was much ado about the "Navigator in a closet" designed by scientists at MIT. Today, thanks to MEMS accelerometers and gyros, the '16350 offers the prospect of a "Navigator in a matchbox."

tronics revolution drove the creation of smaller, lighter, and lower-cost designs. Instead of thousands of pounds and millions of dollars stuffed in a “closet,” advances (e.g., high-performance microprocessors and ring-laser gyros) led to the creation of “strapdown” (i.e., non-gimballed) navigation units that cut the size, weight, and cost by a factor of 100.^[3]

NAVIGATOR IN A MATCHBOX

The navigation recipe calls for accelerometers, gyros, and micros. Analog Devices makes accelerometers, gyros, and micros. Get the picture (see Photo 2)?

I’ve previously covered Analog Devices’s MEMS accelerometers (“XLR8R: Working with Accelerometers,” *Circuit Cellar* 107, 2005) and gyros (“Spin Control,” *Circuit Cellar* 161, 2003), but here’s a quick refresher.

The accelerometer is pretty straightforward. It’s basically a variable capacitor with two fixed plates and a central plate suspended on micromachined springs. Acceleration in the sensitive axis moves the sprung central plate relative to the fixed plates, causing a change in the differential capacitance. A precision signal chain translates the tiny change in capacitance into a high-level “g” output.

The gyro in Figure 1 is even cleverer, if less intuitive. It relies on the Coriolis effect, the same phenomenon that causes storms to spin in different directions north and south of the equator. Here’s a more intuitive example to literally get a feel for what’s going on. Take a partner along to a park or playground with a merry-go-round. Get on the merry-go-round and step back and forth (i.e., oscillate) from the center to the outer rim. If the merry-go-round is stationary, you won’t feel any difference in lateral acceleration at the center or the rim. Now, have your partner spin the merry-go-round while you keep “oscillating” and you’ll notice the lateral acceleration you feel changes as you move outward and inward. Furthermore, the degree of change varies with the rate of rotation. That’s just the way Analog Devices’s MEMS

gyros work (see Figure 2).

MIN PINS

There’s a lot going on under the hood of the Analog Devices ADIS16350, but it’s all hidden behind a mild-mannered facade of just 24 pins (see Figure 3). Connection is via a short flex cable that emerges from the module’s base and terminates in a 2 × 12 1-mm pitch header. With a 2,000-g shock survivability spec, the module itself has hefty brackets for screw mounting to keep it in place when the ride gets rough.

More than half the pins are devoted to power (5 V and ground, three pins each) or “No Connects” (nine pins). That means there are really only nine signals your design needs to care about.

I’m going to assume *RST is pretty much like any other chip’s reset input, since the datasheet has little to say about it. The specs reveal that the initial power-up delay is 150 ms, so don’t expect to rush out of the starting gate.

Most of the action will center on the four pins (*CS, SCLK, DOUT, and DIN) comprising the SPI. Transfers, framed by *CS are 16 bits, each transferred on the falling edge of SCLK with the exception of the first bit (the MSB, i.e., D15) that goes out with the falling edge of *CS. With separate DIN and DOUT lines, the interface is capable of full-duplex transfers (i.e., data can be both written and read on each clock).

A write opera-

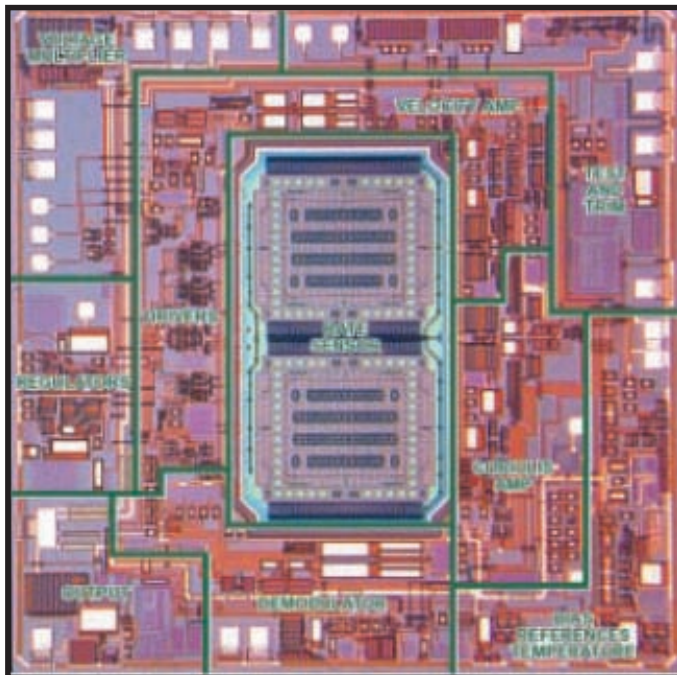


Figure 1—For many years, MEMS technology seemed stuck in the “hype and hope” phase. But now, high-volume production of devices, such as this Analog Devices gyro, prove that MEMS has truly come of age.

tion uses a 2-bit write opcode (“10”), a 6-bit register address, and 8 bits of write data. Because the registers are 16 bits wide, a register write sequence requires two frames.

A read operation starts with a 2-bit read opcode (“00”) and the 6-bit register address. For an initial read, the final 8 bits of incoming data that finish the frame should be ignored. Instead, the entire 16 bits of read data are returned during the next frame. Although the initial read requires two frames, thanks to the full-duplex nature of the interface, the overhead is reduced for streaming reads (i.e., two back-to-back reads require only three frames, not four).

The timing of transfers is rather

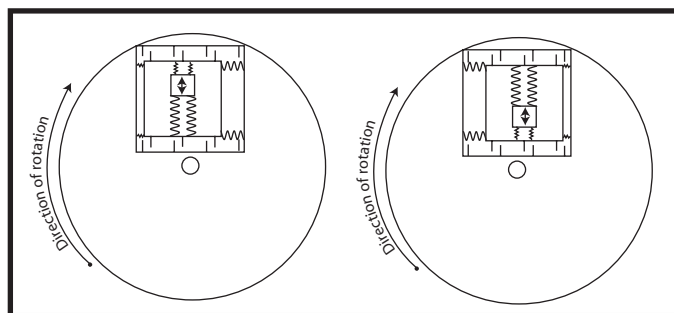


Figure 2—A MEMS gyro is similar to a MEMS accelerometer, except the “sprung mass” oscillates. The rate of rotation is measured by the change in lateral acceleration detected as the mass oscillates towards and away from the axis of rotation.



Go Wireless Go Rabbit®

Check out our new Wi-Fi and ZigBee® core modules — the latest additions to our pin-compatible family.

RCM4510W
ZigBee RabbitCore®

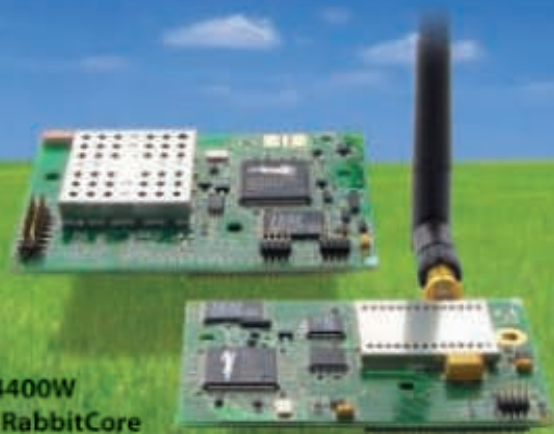
\$72 (qty. 100)

Wireless Development Kits

Only \$199 (reg. \$299)

RCM4400W
Wi-Fi RabbitCore

\$99 (qty. 100)



Applications

- Industrial Control
- Remote Terminal Unit (RTU)
- Building Automation
- Data Acquisition

RCM4500W Family

- ZigBee/802.15.4
- 49 GPIO
- 6 Serial Ports

RCM4400W Family

- Wi-Fi/802.11
- 35 GPIO
- 6 Serial Ports



2000 Spafford Street, Davis, CA 95618 Tel 530.757.8400

Order Your Kit Today

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

www.rabbitkits-wireless.com

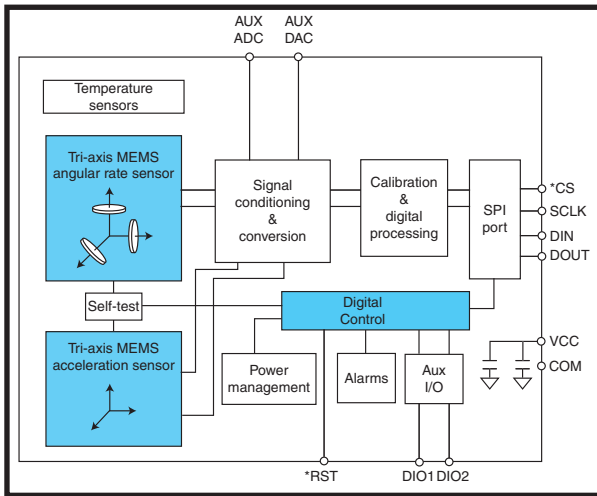


Figure 3—A block diagram of the ADIS16350. Start with some accelerometers and gyros, throw in a dash of “digital control” (i.e., an MCU), and voila, you’ve got an ADIS16350 inertial measurement unit (IMU) and the key building block for a new generation of navigation applications.

flexible and should accommodate any MCU, whether one with a built-in hardware SPI or a lowly bit-banger. The main items to note are that the ‘16350 has a two-speed transmission (“fast” and “normal”) and there are some key specs that constrain transfer timing.

Whether you use Fast or Normal mode is a choice that will be determined by two aspects of your design. The first is the bandwidth (i.e., what sampling rate your application requires). The overall sample rate is quite programmable, from a high of almost 1 kHz to less than 1 Hz. Right around 150 Hz is the dividing line between Fast and Normal modes. The naturally related consideration is power consumption for the 5-V module. For Fast mode, the typical spec is 57 mA, while for Normal mode, it’s 33 mA. Both of these are pretty high, so for battery-powered applications, you’ll likely want to take advantage of the module’s “Sleep” mode. It cuts power consumption to just 500 μ A (independent of

the Fast or Normal mode selection) while retaining the ability to wake up in just 3 ms, quite fast compared to the aforementioned 150-ms power-up delay.

Once you’ve made your choice of Fast or Normal mode based on bandwidth and/or power consumption requirements, you’re looking at a maximum SPI clock frequency of 2 MHz or 500 kHz, respectively, although even the latter is pretty “fast” to my way of thinking.

The other SPI timing consideration revolves around “data rate” and “data stall” specs, which

also vary depending on the Fast or Normal mode selection. The data rate spec defines the minimum time between frames, while data stall specifies the minimum time between the last bit of a frame and the beginning of the next one. For example, in Fast mode, the data rate spec calls for a minimum of 40 μ s between falling edges of *CS and a minimum of 9 μ s from the last SCLK of a frame to the beginning of the next frame. For Normal mode, the specs are 160 μ s and 75 μ s, respectively. You can see that when it comes to the overall SPI throughput, these specs are the limiters, not the SPI clock itself. For instance, while the SPI clock can run up to 500 kHz in Normal mode (i.e., 2 μ s/bit), the data rate spec of 160 μ s per 16-bit frame says overall throughput is limited to 100 kHz (i.e., 10 μ s/bit). In any case, the SPI is certainly fast enough that it won’t be getting in the way.

Moving on, the ‘16350 includes an ADC and a DAC for general-purpose use, with a pin devoted to each (AUX ADC and AUX DAC, respectively). Both feature 12-bit resolution and otherwise seem pretty middle of the road spec-wise, with one caveat being the voltage range is limited to 0 to 2.5 V for both units.

Finally, there are two digital I/O lines, DIO1 and DIO2. In addition to general-purpose use, the lines can be configured for special functions. One option is to devote one line as a “data ready” output that signifies the completion of an internal sampling cycle. This is natural to drive an interrupt input or status polling bit of an attached controller, all the more so since polarity is programmable.

Register name	Function
ENDURANCE	Flash memory write count
SUPPLY_OUT	Power supply measurement
XGYRO_OUT	X-axis gyroscope output measurement
YGYRO_OUT	Y-axis gyroscope output measurement
ZGYRO_OUT	Z-axis gyroscope output measurement
XACCL_OUT	X-axis acceleration output measurement
YACCL_OUT	Y-axis acceleration output measurement
ZACCL_OUT	Z-axis acceleration output measurement
XTEMP_OUT	X-axis gyroscope sensor temperature measurement
YTEMP_OUT	Y-axis gyroscope sensor temperature measurement
ZTEMP_OUT	Z-axis gyroscope sensor temperature measurement
AUX_ADC	Auxiliary analog input data
XGYRO_OFF	X-axis gyroscope bias offset factor
YGYRO_OFF	Y-axis gyroscope bias offset factor
ZGYRO_OFF	Z-axis gyroscope bias offset factor
XACCL_OFF	X-axis acceleration bias offset factor
YACCL_OFF	Y-axis acceleration bias offset factor
ZACCL_OFF	Z-axis acceleration bias offset factor
ALM_MAG1	Alarm 1 amplitude threshold
ALM_MAG2	Alarm 2 amplitude threshold
ALM_SMPL1	Alarm 1 sample period
ALM_SMPL2	Alarm 2 sample period
ALM_CTRL	Alarm source control register
AUX_DAC	Auxiliary DAC data
GPIO_CTRL	Auxiliary digital I/O control register
MSC_CTRL	Miscellaneous control register
SMPL_PRD	ADC sample period control
SENS/AVG	Dynamic range/digital filter control
SLP_CNT	Counter used to determine length of Power Down mode
STATUS	System status register
COMMAND	System command register

Table 1—The ‘16350’s onboard flash MCU makes things easier by consolidating all the action taking place behind the scenes into a straightforward register map accessed via a simple SPI.

Another option has one or both digital I/O lines configurable as “alarm” outputs with very programmable trigger conditions. Each of the two alarm outputs can be tied to any of the primary functions including the X, Y, or Z axis of the accelerometers and gyros. Other trigger options include the temperature sensors (each gyro has one), a power supply monitor, and the AUX ADC input. Going beyond simple static threshold comparisons, the trigger logic has a “rate of change” option (i.e., triggers on delta rather than an absolute threshold) and a “filtering” option that uses an average of prior readings as the basis for threshold comparison. As with the “Data Ready” mode, the alarm output polarity is programmable.

The software interface to the '16350 comprises 30 or so registers, all 16 bits wide (see Table 1). The majority of the action centers on less than a dozen of them, including the three (i.e., X, Y, and Z) gyro outputs, three gyro temp sensor outputs, and three accelerometer outputs. The gyros feature programmable scaling of 75, 150, and 300 degrees per second while the accelerometers are ± 10 -g units and both offer 14 bits of resolution. In addition, there's a register that monitors the power supply and one that tracks the aforementioned general-purpose ADC input.

Another couple dozen registers handle setup and control. Six of them are gyro and accelerometer offset correction registers. As calibrated at the factory, these come initialized to 0, but they can be adjusted during operation to fine-tune for a particular installation. The other registers control the aforementioned general-purpose DAC and digital I/O, alarm configuration, and Sleep mode.

Note that the module transfers the register contents from flash memory to SRAM at reset. Any changes you make will be lost at power down unless you commit them to flash memory using a “flash update” command. Although few applications will likely run up against the flash write endurance spec (10,000 cycles), for those that might, a nice touch is an ENDURANCE register that tracks the number of flash writes. As well, I noted

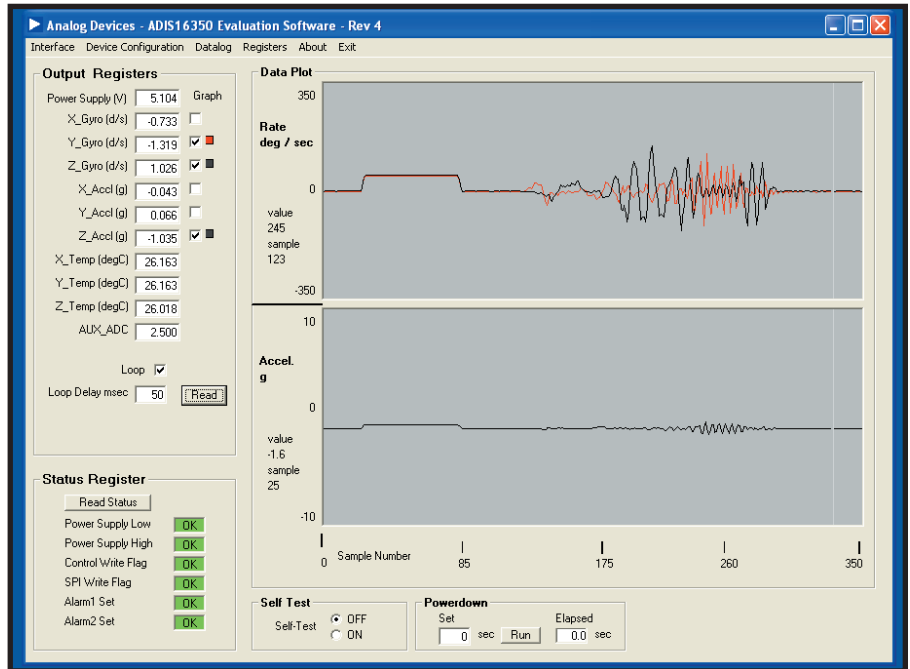


Photo 3—The software that comes with the '16350 evaluation kit does a good job configuring the module options (such as the self-test you can see here at the beginning of the sample period) and tracking acceleration and rotation rates. As for the software and hardware needed to turn these inertial basics into a full-featured inertial navigation system (i.e., position and attitude tracking), that's up to you.

that the flash memory data retention spec is 20 years, quite a bit better than the typical 10 years I consider marginal for embedded applications.

FLY BY WIRE

I had a chance to play with the '16350 using a preliminary version of the evaluation kit, a two-board lash-up using, believe it or not, a Centronics parallel interface. Fortunately for me, I still have a PC with this dying breed of connector and I didn't even need to fiddle with any BIOS settings. Fortunately for you, Analog Devices informs me that by the time you read this, there should be an updated board with a USB interface.

Come to think of it, someone should whip up a battery-powered wireless version. As I was waving the thing around in an inertial frenzy, and thereby tying the cables (Centronics and power) in knots, I concluded that wires definitely cramp the '16350's “six degrees of freedom” style.

Despite the old-school wiring, I found the PC software that comes with the kit to be informative and work well. The main screen consists of a line chart that enables tracking the accelerometer and gyro outputs and optionally capturing the data to a

file (see Photo 3). Other screens handle configuration (e.g., gyro range, sample rate, and alarm conditions), calibration, and the auxiliary I/O (ADC, DAC, and digital). The only suggestion I have is that they make the vertical axis scale of the line charts adjustable so you can better see small transitions.

MAKE VS. BUY?

Today the price quote for a '16350 is \$275 (1,000-piece price). There is no arguing that in the world of navigation systems that's a bargain indeed. Nevertheless, it's a lot for a “chip,” especially considering that it's way more than the bill of materials for the collection of parts (i.e., three-axis accelerometer, three gyros, and a flash MCU) under the hood. It's tempting to think this is a situation where it's better to make versus buy.

Just don't overlook the fact that testing and calibrating something with six degrees of freedom can lead to six degrees of headaches. To really put the thing through its paces, you need quite a fancy test fixture to accurately control, characterize, and calibrate across the entire range of the device's capability.

That's not even considering temperature compensation, a huge challenge

for extended temperature range applications, and a big complication for test and calibration. To that end, Analog Devices offers a -40° to 85°C version of the module (the ADIS16355, \$359) that features 10 times better stability (bias and sensitivity) across the temperature range.

Whether you decide to roll your own or let Analog Devices do it for you, there's still plenty of work to do. Remember, the '16350 is an inertial measurement unit (IMU). All it does is provide an instantaneous snapshot of the forces acting on it. Presuming you're actually interested in tracking position and attitude, what you really need is an inertial navigation unit (INU).

Indeed, early on, I was wondering why Analog Devices didn't go all the way and offer a true INU. Instead of acceleration and rotation rates, the module would track all that and compute the current location and attitude for you. But the more I got into it, the more I realized that while "M" and "N" may be right next to each other in the alphabet, turning an IMU into an INU isn't easy.

The main difficulty is that the requirements for an INU are very application specific, and thus it's hard to define a generic solution. Since open-loop "dead reckoning" is only accurate over the short term, an optimal INU design for a particular application must take advantage of any and all opportunities to fine-tune the navigation solution.

For example, a robust INU design might find an IMU combined with a GPS, each backstopping the other. When the GPS drops out, the IMU takes over. Otherwise, the incoming GPS data is used to continually cross-check and calibrate the output of the IMU.

But depending on your application, there may be other clever and cheaper ways to obtain such synergistic cooperation. For instance, in a vehicular application, the simple ability to detect if the vehicle has stopped (e.g., wheel sensor) can be very helpful. It provides a reality check (i.e., "velocity fix") for the INU, which due to error accumulation, may otherwise think the vehicle is still moving.

In other words, the right navigation solution for a particular application will "take data from as many different sources as is economically practicable, and combine them in the best way possible."^[3]

HAVE NAV, WILL TRAVEL

Like breakthrough sensors in the past, I expect the '16350 will usher in a new round of applications. Once again, progress will be on two fronts.

Surely the low price of the unit compared to traditional solutions will fuel more demand for existing applications. For example, real aircraft have INUs, and now model aircraft can too.

Prospects for enabling yet unforeseen "blue sky" applications are even more exciting. How about an underwater navigation system for divers (GPS doesn't work underwater)? Or an autopilot for a hang-glider? Or a golf swing analyzer? Or for that matter, how about an "active" golf club that turns any duffer into a Tiger on the links?

Let your imagination run wild. It's only a matter of when, not if. ☐


Tom Cantrell has been working on chip, board, and systems design and marketing for several years. You may reach him by e-mail at tom.Cantrell@circuitcellar.com.

REFERENCES

- [1] Time, Inc., "Here to There, Accurately," April 1957, www.time.com/time/magazine/article/0,9171,821124-1,00.html.
- [2] C. Sublette, "Advanced Inertial Reference Sphere," The Nuclear Weapon Archive: A Guide to Nuclear Weapons, <http://nuclearweaponarchive.org/Usa/Weapons/Airs1.jpg>.
- [3] A. King, "Inertial Navigation—Forty Years of Evolution," *iMAR: Systems for Inertial Measuring, Automation and Control*, www.imar-navigation.de/download/inertial_navigation_introduction.pdf.

SOURCE

ADIS16350 Inertial measurement unit
Analog Devices, Inc.
www.analog.com




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

USB/ETHERNET DAO

LabJack UE9



Available now for only ...
\$429 qty 1

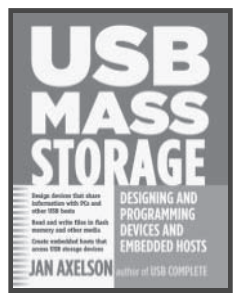
USB/Ethernet Data Acquisition & Control

- * USB 2.0/1 and Ethernet
- * 14 analog inputs (12- to 16-bit)
- * Stream input data up to 50 kHz
- * Uses LabVIEW, VB, LabVIEW, etc.
- * Includes DAOFactory Express
- * Operates from -40 to +85 deg C
- * 2 analog outputs (12-bit)
- * 22 digital I/O
- * Up to 2 counters (32-bit)
- * Up to 8 timers
- * Approx. 3" x 7" x 1"

LabJack Corporation, Colorado, USA
info@labjack.com, (303) 942-6228

www.labjack.com

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

IDEA BOX

THE DIRECTORY OF PRODUCTS AND SERVICES

AD FORMAT: Advertisers must furnish digital submission sheet and digital files that meet the specifications on the digital submission sheet. **ALL TEXT AND OTHER ELEMENTS MUST FIT WITHIN A 2" x 3" FORMAT.** Call for current rate and deadline information. 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) 875-2199.

The Suppliers Directory at www.circuitcellar.com/suppliers_dir/ is your guide to a variety of engineering products and services.



phyCORE[®] OEMable Single Board Computers

XScale: PXA270, PXA255

ARM: LPC3180 (ARM9); LPC22xx, LPC229x, AT91 (ARM7)

PowerPC: MPC5554, MPC5200B, MPC5655, 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

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



Start your own robot business!



Tell us about yourself and we'll tell you about the plan at robotbusiness@smartrobots.com

Low Cost PGM/DEMO Kits



USB-DONGLE

The **USB-DONGLE** and **Derivative Boards** allows quick and easy ICPISP programming of many popular microcontroller families such as the LPC9xx, ARM7 LPC2xxx and 89V52X2 from NXP Semiconductors. The kit includes a Virtual COM Port Driver that allows hex files to be downloaded and programmed using Flash Magic or other common utilities. The unit also provides a low cost platform for testing or prototyping of simple microcontroller based designs (blink an LED, measure I/C, test a Timer or PCA output). Low cost **Derivative Boards** are available for many different microcontrollers from NXP. Please consult our website for details.

Derivative Boards



LPC9103 HVSON10



ARM7 LPC2103 LQFP48

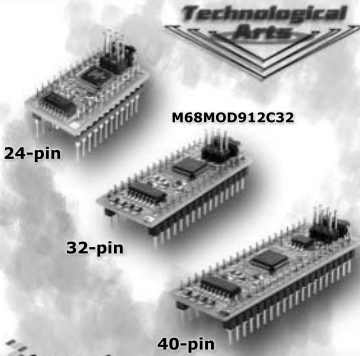
XA Development Kits, I2C, MDIO & SPI tools in stock, visit www.teamfdi.com for details.



www.teamfdi.com
 VISA/MC/Amex

Future Designs, Inc.
 2702 Triana Blvd
 Huntsville, AL 35805
 (256) 883-1240
 Fax (256) 883-1241

Freescale 9S12C microcontroller DIP modules



Technological Arts

M68MOD912C32

24-pin

32-pin

40-pin

freescale
 Alliance Member

www.nanocore12.com
 Toll-free: 1-877-963-8996

7 amazing advances in 1 microcontroller programmer

- #1: Create cash** thru per-copy licenses
- #2: Secure** delivery of your intellectual property to customers
- #3: No rip-off copies** by subcontractors
- #4: EZ archive.** Your code stays onboard
- #5: Tidy.** Plug in, no wires
- #6: It can fly.** Airmail upgrades to customers
- #7: Low cost.** From as little as **\$32** each at Mouser & Digikey
- #8! Ready for business** with HexWax, the firmware publishing service



Actual size - patents pending



TEAclipper

www.flexipanel.com www.hexwax.com

CHINA PCB SUPPLIER

 **DIRECT SALE FROM CHINA**
 **PROTOTYPE TO PRODUCTION**

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



web: <http://www.pcbcart.com>
 email: sales@pcbcart.com
 Tel: +86-571-87028889
 Add: No. 2 Haining Road,
 Hangzhou, PR China

WWW.PCBCART.COM

www.can232.com

Only \$199


CAN232 Features:
 Free sample programs
 9-18VDC supply via CAN
 Timestamp in mS
 Small size 2.7" by 1.2"
 100% Bandwidth up to 120Kbit
 Both 11 & 25 bit ID support
 32 Message Receive FIFOs
 Works up to 1Mbit CAN
 Simple ASCII protocol
 Supports RTR Frames
 Max 230Kbaud RS232
 Firmware upgradeable
 No drivers needed
 OS independent
 CE Approved

CANUSB Features:
 Free ActiveK component
 PC, MAC & Linux support
 Both 11 & 25 bit ID support
 Simple CAN logger included
 Free Threaded Windows DLL
 Firmware upgradeable via USB
 Sample programs in C, C++, VB,
 Delphi, C#, PureBasic etc.
 No need for external power
 Works up to 1Mbit CAN
 Supports RTR Frames
 USB 2.0 Full Speed
 Free USB drivers
 CE Approved

Only \$154 \$129

www.canusb.com

Instant LCD



esLCD-002 - \$199
Versatile Programmable Module


- Serial/PC/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

EARTH.LCD.COM We Make **LCDs** Work.™

Product Development
Industrial Design
Electrical & Mechanical Engineering
Prototyping and Manufacturing



Quick-turn PCB
Prototype Assembly
As little as one board!


Engineering Your Path to Profit

IMET Corporation

2614 Ford Road, Unit B • Bristol, PA 19907
 Tel: 215-788-5256 • 866-435-4907
 Fax: 215-788-5362
www.imetcorporation.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



Hands-on Training
 for Rabbit Developers

Learn More at
Rabbit-U.com

1,600+ Products
www.DesignNotes.com

Project Cases
Connectors & Adapters



100's of
Pre-designed Electronic Kits

Test Equipment
 USB PC 2 CH
 60MHZ Oscilloscope
 \$329.99 **+ free gift!**

DesignNotes.com
 Affordable Prototyping Products

1-800-957-6867

Customer Design Front Panels


We work with your CAD program

Expert in Quality
Expert in Service
Expert in Material
Expert in Price

Please contact us.
 Email: info@cam-expert.com
 Web: www.cam-expert.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

LV- MaxSonar

Ultrasonic Ranging is EZ

actual size



- Low power!
2.5V-5.5V,
current 2mA
- Tiny size!
less than
1 cubic inch
- Low cost!

- Choice of beam patterns.
- Easy interfacing. Serial, pulse width, & analog voltage outputs.
- Reliable and stable range measurement.
- No dead zone. Detect objects up to the front sensor face.

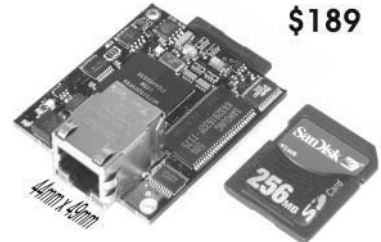
For more info please visit us at
www.maxbotix.com

USB Host Stack

- USB 2.0
- EHCI, OHCI, UHCI, ISP116x, ISP1362, ISP176x, ARM, ColdFire
- Hub, Mass Storage, Modem, Mouse, Keyboard, Printer, Serial Converter
- Low Cost, Royalty-Free
- Full Source Code
- Standalone or RTOS
- Device and OTG Available

 **Micro Digital Inc**
RTOS Innovators

www.smxrtos.com/usb



\$189

Tiny Linux controller

16MB fast SDRAM 10/100 Ethernet
64MHz Coldfire MCU 3 serial ports (RS232)
4.5 MB flash memory 1 CAN port
SD card socket (to 2GB) LCD/KPD port

Eclipse/CDT dev. env. Free serial debugger

55 I/O pins & capabilities to burn...

www.steroidmicros.com

Instant Wireless Dev. Kit (433 MHz)

www.radiotrix.com

RK-433-RC
DEVELOPMENT KITS
from

RADIOTRIX



433MHz Wireless Remote Control

- ASK/ OOK
- 2 - 12 V solutions
- Antenna design included
- Features MicroChip KEELOQ™ (HCS300) & (HCS512)
- Complete transmitter & receiver development platform
- Multi-Receiver Unit works w/ all Radiotrix receivers
- Kit contains 4 receivers and 2 transmitters (SecureFOB™)
- All design files available through Radiotrix sales channels
- SecureFOB™ supports 3 push button controls
- Low cost MSRP \$99

AVAILABLE NOW!
(RK-433-RC)

ORDER HERE!

COMPONENTS
Designing in
Electronic
Components

www.dcomponents.com 1-802-752-4321

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



www.intrepidcs.com

GHz Bandwidth Sockets for DSP's in BGA



Sockets for dozens of DSP chips from TI, Analog Devices, and other manufacturers.

- Pitch - 0.3mm to 1.27mm - BGA, QFN (MLF)
- Bandwidth to 10+ GHz
- Smallest Socket Footprint in Industry
- Optional 500,000 insertions
- Heatsinking to 100 watts

We design custom sockets to your requirements including heat sink option. Request our Free Catalog/CD today.



Ironwood Electronics

1-800-404-0204

www.ironwoodelectronics.com

Program in C and develop your PIC® external memory projects ... in a flash!

\$624

Now \$604*

PIC18F8722
Development Kit & PCWH
Windows IDE Compiler.

[discount code FLASH18]

*[Offer good until November.30.07. Shipping not included. Not valid with any other offer.]

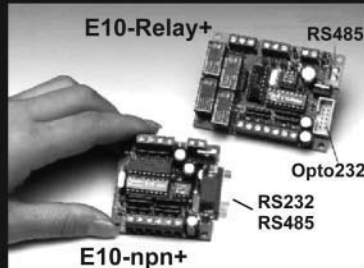
www.ccsinfo.com/flash
262.522.6500 x35

CCS



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/ccl.htm

Tel: 1-877-874-7527 - PLC specialist since 1993

TRI
Triangle
Research

Best Home LED



- Newest technology
- Brightest, most efficient LEDs
- Energy saving
- Long life
- Cool operation

www.besthomeLEDlighting.com

for a wide selection of LED bulbs,
low prices and free shipping!

Add a color touch interface to your embedded product!



- High level RS232
- Low cost interface
- Easy to program
- In stock!

Add color graphics to any 8/16 bit embedded system. Easy, fast and flexible. Up and running in hours!

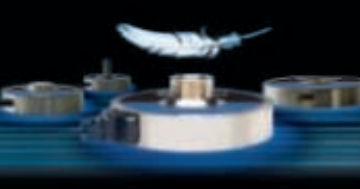
REACH
TECHNOLOGY, INC.

www.reachtech.com • 510-770-1417

842 Reggo Avenue • Fremont, CA 94538

Loadstar SENSORS

Introduces
Capacitive Load Sensors with
True USB connectivity



Integrated signal conditioning
Digital USB or Analog 0-5 V output
Accuracies - 0.25% to 0.025% of FS
Rugged stainless steel construction
Temperature compensated
Easy mounting features built in

www.loadstarsensors.com
650.938.4282 | info@loadstarsensors.com

Kanda.com


Manufacturer of the original
STK200 & STK300
AVR development kits
Launch the latest addition to the Kanda
tools range:
AVRISP-U



Alone as an AVR
USB port In System Programmer
or with the
STK200 & STK300 range
of development kits.

For all your embedded tools requirement
contact:
www.kanda.com
sales@kanda.com

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

32io SERVER



control & monitor
anything anywhere
over the internet

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.

Free one time setup!
(subject to approval)


W/5 double side PCB
2 weeks

Tel: 800.695.9342
sales@mylydia.com

www.mylydia.com

586-Generation Industrial Control

586Drive+P300 Ethernet/TCP, 24-bit ADC, DAC, HV I/O,
CF, 300+ HV I/O with screw terminals.



- 586 Drive (control board) + P300 (expansion board)
- AMD SC520 processor, program in C/C++
- 4 RS232/485, ADC, DAC, Solenoid Drivers, OPTO
- CompactFlash and FAT16 file system support
- Hardware TCP/IP stack for 100M Base-T Ethernet

50+ Low Cost Controllers with ADC, DAC, solenoid drivers, relays, PC-104, CF, LCD, DSP motion control, 10 UARTs, 300 I/Os. Custom board design. Save time and money.

TERN INC. 1724 Picasso Ave., Suite A, Davis, CA 95616 USA
Tel: 530-758-0180 • Fax: 530-758-0181

www.tern.com • sales@tern.com

VISA MasterCard

NEW

RS232 to TCP/IP

- TCP/com[™] v2.0, RS232 to TCP/IP software. Plus TCP/IP to RS232.
- WinWedge[™] RS232 or TCP/IP data direct into any Windows app. - Excel, Access, etc.

TALtech

Free 30 day evals at www.taltech.com

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.

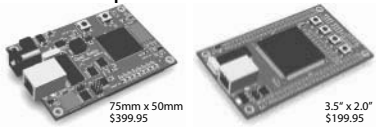
2007 marks our 11th year of selling CAN interfaces in over 30 countries!

\$249
USD.



Zanthic Technologies Inc.
403-878-2202
www.zanthic.com

High-speed USB 2.0 + Xilinx FPGA Complete Software API



75mm x 50mm \$399.95 3.5" x 2.0" \$199.95

<p>XEM3010-1500P:</p> <ul style="list-style-type: none"> • 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 	<p>XEM3001:</p> <ul style="list-style-type: none"> • 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

CUSTOM MEMBRANE KEYBOARDS / SWITCHES



- 1 TO 2 WEEKS TURNAROUND
- VERY COMPETITIVE PRICING
- Ex.: (5) 4-switch keyboards for \$395.00
- PCB backed switches
- Custom metal backplates/assemblies
- Electronic assemblies/graphic overlays
- Electronic file transfer capabilities

Picofab Inc.

4780B Blvd. Henri-Bourassa
Charlesbourg, Quebec, Canada G1H 3A7
Tel: (418) 622-5298 • Fax: (418) 622-9996
Email: sales@picofab.net

FAT 12/16/32 FILE SYSTEM

- DOS/Windows Compatible
- USB Flash Disk
- USB Floppy & Hard Disk
- SD/MMC
- CompactFlash, ATA/IDE
- DiskOnChip
- NAND & NOR Flash
- 10KB RAM / 25KB ROM typical
- Low Cost, No Royalty
- Full Source Code

www.smxrtos.com

Micro Digital Inc
RTOS Innovators

800.366.2491 sales@smxrtos.com

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

Front Panels?

Download the free Front Panel Designer to design your front panels in minutes

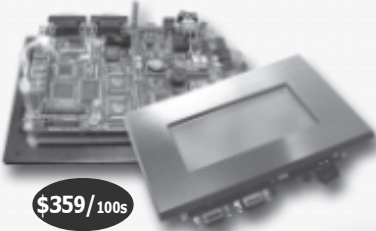



Unrivaled in price and quality for small orders

Order your front panels online and receive them just in time

www.frontpanelexpress.com

QScreen™ - Low Cost Versatile Instrument Controller



\$359/100s

- Touchscreen Operated GUI
- 128x240 CCFL- Backlit Graphic Display
- Programmable in C or Forth
- Hundreds of Screens, Buttons, & Menus
- A/D and Two RS232/485 ports
- 8 Timer-Controlled Digital I/O Lines
- Up to 1 MB Flash, 512KB RAM
- Wide Selection of Plug-in I/O

Mosaic Industries Inc.
tel: 510-790-1255 fax: 510-790-0925
www.mosaic-industries.com

Full Speed CAN USB Adapter

Simple configuration & use



+1 630-245-1445
Naperville, Illinois USA

www.c-a-n.com

gridconnect



ANYONE

Can Now Easily
Hand Solder Surface-
Mount Components!

Even A 10 Year Old!

www.schmartboard.com

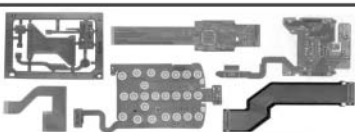


Weather Instruments

for PCs



www.agelectronica.com



1-6 Layers Flex PCB
High Quality
1-2 Weeks Delivery
Start at Only \$299



Custom Membrane Keyboards
PCB/FPC Backed, Built-in LED
Start at \$245

EZPCB www.ezpcb.com
sales@ezpcb.com

Flashlite 186

- 186 processor = 33 MHz
- DOS w/ Flash File system
- 44 Digital I/O lines w/ CPLD
- Console / Debug Serial Port
- 7-34V DC or 5V DC power
- 2 Serial Ports
- Accepts 8MB DiskOnChip
- 2 16-bit Timers
- 512K DRAM & 512K Flash
- Watchdog Timer
- Expansion options with Peripheral Boards

\$69 QTY 1

\$99 Development kit includes:

- Flashlite 186 controller
- Borland C/C++ ver 4.52
- FREE Email Tech Support
- Serial Driver library
- AC Adapter and cable

Call 530-297-6073 Email sales@jkmicro.com
On the web at www.jkmicro.com

JK microsystems

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

PROTOTYPE CIRCUIT BOARDS

\$85 U for two 5" x 6"
S 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

PRINTED CIRCUIT BOARDS

QUALITY PRODUCT

FAST DELIVERY

COMPETITIVE PRICING

- Aluminum Backed PCB
- Single & Double sided
- SMOBC/RoHS
- LPI mask
- Through hole or SMT
- Nickel & Gold Plating
- Routing or scoring
- Electrical Testing
- Artwork or CAD data
- Fast Quotes
- Flex Circuits

PROTOTYPE
through
PRODUCTION

SPECIAL OFFER:

10 pcs (3days) 1 or 2 layers \$249

10 pcs (5days) 4 layers \$695

(up to 30sq. in. ea.) includes tooling, artwork, L.P.I. mask & legend

PULSAR, INC

9901 W. Pacific Ave. Franklin Park, IL 60131 • Phone 847.233.0012
Fax 847.233.0013 • www.pulsar-inc.com • 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.

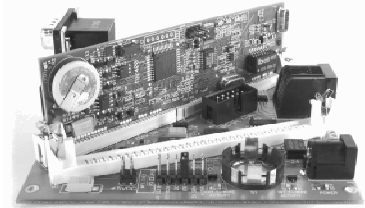
HIDmaker FS Combo: Only \$599.95

DOWNLOAD the HIDmaker FS Test Drive today!

www.TraceSystemsInc.com
301-262-0300

TStik!™

Rugged TINI Java™ Module
with 10/100 BaseT



TStik is a ruggedized TINI400 chipset in the familiar SIMM72 form factor. Upgrade most DSTINI1 (TINI390) systems or use our new TILT socket boards (TILT Pro is shown above).

TStik is about \$100, and sockets start at under \$60.

SYSTRONIX®

full details at www.TStik.com

WIRELESS RS-232

WCSC (Wireless 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.wcsnet.com>
sales@wscnet.com (281)360-4232

Controllers

NEW ET-AVR Stamp Module



ONLY \$19.90

- Includes ATmega64 Microcontroller
- 64kbytes Flash Program Memory
- Up to 53 I/O Ports
- 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



Complete SDK Online at: www.ontrak.net

FEATURES

- 8, 5-AMP relay outputs
- 8, ISOLATED digital inputs
- Port Powered, Aux 5VDC output \$189.00 QTY 1

Other Models:

- ADU200- 4 Channel Version with RS232 \$139.00
- ADR218- Solid-State Version 8 Channel \$225.00
- ADU100- 3 CH, 16-Bit ISOLATED Analog Inputs, PGA, 4 digital I/O, RS232 and 5 AMP Relay Output \$199.00

ONTRAK CONTROL SYSTEMS INC.
PH: (705) 671-2652 Fax: (705) 671-6127

www.ontrak.net

Order online at:
www.melabs.com

microEngineering Labs, Inc.

Phone: (719) 520-5323
Fax: (719) 520-1867

Development Tools for PIC® Microcontrollers

Box 60039
Colorado Springs, CO 80960

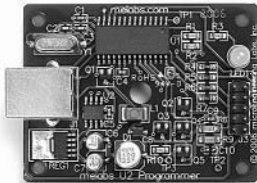
USB Programmer for PIC® MCUs

\$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

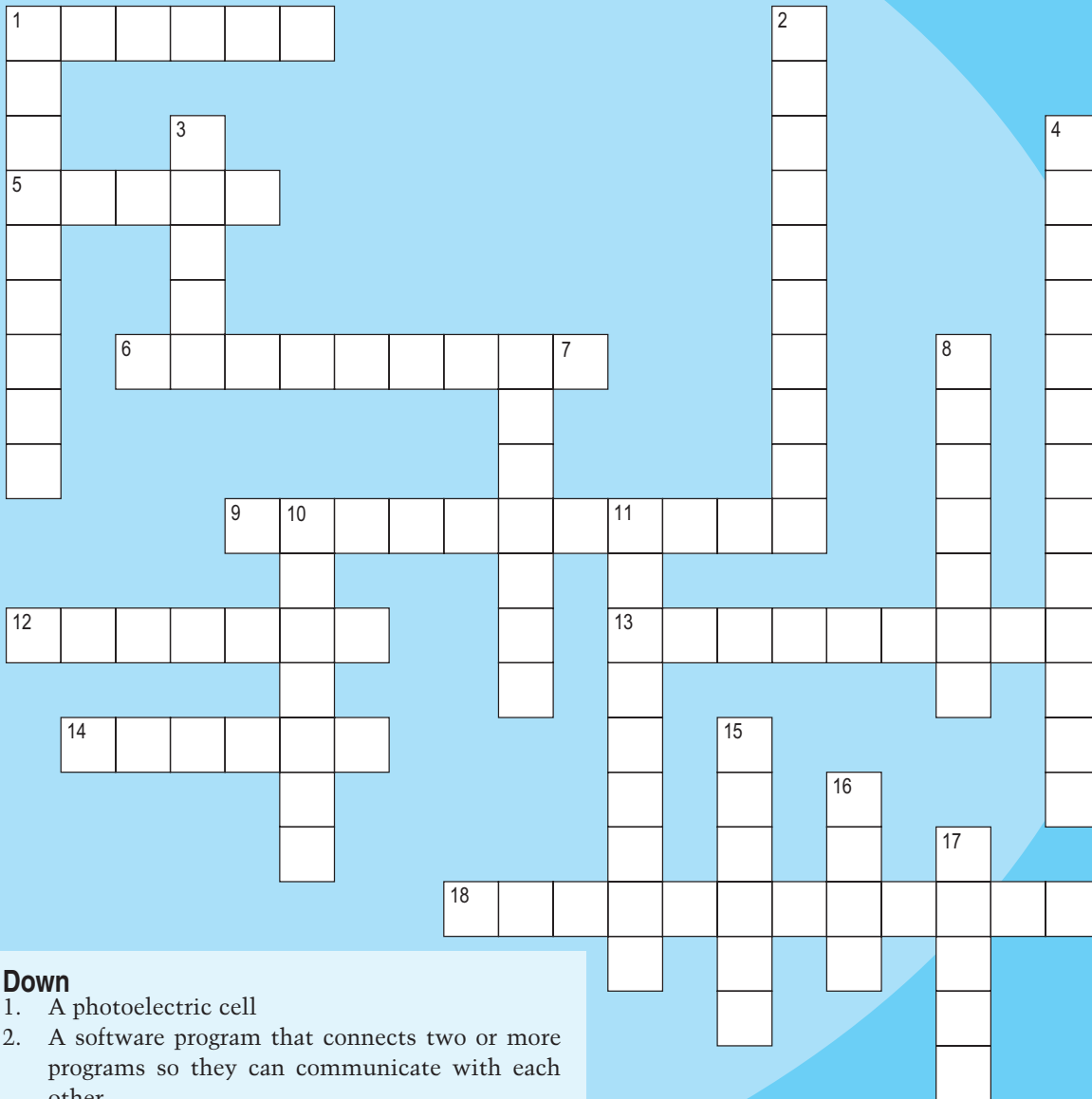
PICBASIC PRO™ Compiler \$249.95

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



Down

1. A photoelectric cell
2. A software program that connects two or more programs so they can communicate with each other
3. The dark center of a sunspot
4. Teasing on the 'Net
7. An object in space that moves in front of or through the shadow of another
8. Ctrl
10. The horizontal angle (measured in degrees) between the direction of a fixed point and the direction of an object
11. Alt
15. Computer or program that controls a network's resources
16. Desk for assistance
17. The nickname of a popular Star Wars villain

Across

1. German physicist (1858–1947); Quantum theory
5. Type of small, portable drive
6. Preferred URLs
9. Cap
12. A group of tools used to create and preserve databases
13. The technology to supply voice communications
14. Used to show the pins in an electric connector
18. PV

The answers are available at
www.circuitcellar.com/crossword.

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	3 Embedded World	2 Link Instruments	9, 82 Rabbit Semiconductor
91 AP Circuits	29 ExpressPCB	9,73 Linx Technologies	87 Rabbit Semiconductor
37 ASIX	10, 91 ezPCB	89 Loadstar Sensors, Inc.	88 Radiotronics – DComponents
89 Abacom Technologies	86 FDI-Future Designs, Inc.	39 Luminary Micro	89 Reach Technology, Inc.
73 Ad Hoc Electronics	86 FlexiPanel Ltd.	87 MCC (Micro Computer Control)	17, 41 Renesas Technology
79 Aimtec – DComponents	90 Front Panel Express, LLC	88 MaxBotix Inc.	10 Saelig Co.
91 All Electronics Corp.	92 Futurlec	74 Maxstream	91 Schmartboard
87 Apex Embedded Systems	87 General Circuits, Inc.	48 Microbridge Technologies Corp.	49 Sealevel Systems
7 Atmel	74, 91 Grid Connect	88, 90 Micro Digital, Inc.	11 SEGGER Microcontroller Systems LLC
88 BestHomeLEDLighting	31 HI-TECH Software LLC	5 Microchip	66 Signum Systems Corp.
25 Bitscope Designs	79 HobbyLab, LLC	92 microEngineering Labs, Inc.	86 Smart Robots
21 Bitwise Systems	87 IMET	90 Mosaic Industries, Inc.	61, 92 Systronix
65 CWAV	79 IMAGEcraft	19 Mouser Electronics	90 TAL Technologies
28 CadSoft Computer, Inc.	88 Intec Automation, Inc.	89 Mylydia, Inc.	C3 Tech Tools
87 Cam Expert LLC	88 Intrepid Control Systems	C2 NetBurner	56, 57 Technologic Systems
40 Comfile Technology, Inc.	55 Intronix Test Instruments, Inc.	69 Nurve Networks LLC	86 Technological Arts
88 Custom Computer Services, Inc.	88 Ironwood Electronics	92 Ontrak Control Systems	89 Tern, Inc.
1 Cypress Microsystems, Inc.	64, 91 JKmicrosystems, Inc.	90 Opal Kelly Inc.	71 Tibbo Technology, Inc.
86 DLP Design	33 Jameco	72 PCB-Pool	92 Trace Systems, Inc.
69 DMM Technology Corp.	69 Jeffrey Kerr, LLC	C4 Parallax, Inc.	88 Triangle Research Int'l, Inc.
85 Decade Engineering	89 Kanda.com	86 Phytex America LLC	92 WCSC (Willies Computer Software Co.)
87 Designnotes	42 Keil Software	90 Picofab Inc.	47 Wiznet
61 EMAC, Inc.	85 LabJack Corp.	90 Pioneer Hill Software	15 Wiznet iEthernet Design Contest 2007
87 Earth Computer Technologies	85 Lakeview Research	32 Pololu Corp.	90 Zanthic Technologies, Inc.
27 Efficient Computer Systems	87 Lawicel AB	91 Pulsar, Inc.	
64 Elprotronic	95 Lemos International	8 RD2Tech LLC	

Preview of December Issue 209 Theme: Graphics & Video

SD Card Display Controller

Tile Graphics

LCD Digital Voltage Meter: Talking to the Glass

Solar-Powering the Circuit Cellar (Part 1): The Installation

Build a Learning Remote Control

Programmable Voltage Source

ABOVE THE GROUND PLANE LED Power

THE DARKER SIDE Are You Locked?: A PLL Primer

FROM THE BENCH Collecting Solar Energy

SILICON UPDATE Harvest Time

ATTENTION ADVERTISERS

January Issue 210 Deadlines

Space Close: Nov. 13

Material Close: Nov. 21

Theme: Embedded Applications

**BONUS DISTRIBUTION:
DesignCon; MSC**

Call Shannon Barraclough
now to reserve your space!

860.875.2199

e-mail: shannon@circuitcellar.com

Going Wireless is Easy!



Fourier Data Loggers

- New ZigBee Wireless Data Logging
- Intelligent No Data Loss Transmission
- Portable Long Range Units

USB ZigBee™ Stick

- Wireless Mesh Networking
- New ZigBee Pro Technology
- Low Cost



Smallest RS232 Bluetooth Cable Replacement on the market



47x34x19

Industrial Bluetooth™

- RS232 Cable Replacement
- Long Range
- Low Cost

Multi-Channel 900 MHz Wireless RF Modules

- US & International frequencies available
- High Data Rate
- Low Power



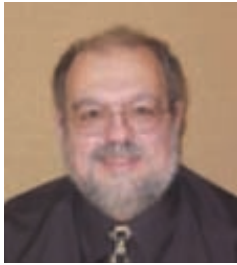
LEMOS

INTERNATIONAL

www.lemosint.com

1-866-345-3667

sales@lemosint.com



PRIORITY INTERRUPT

by Steve Ciarcia, Founder and Editorial Director

Let There Be Light

When kids go back to school in the fall, the first thing teachers usually ask is, "What did you do with your summer?"

"Well, lady, I spent the summer learning everything you never wanted to know about concrete, wind loading, structural steel pipes, and disaster-recovery landscaping." Needless to say, there wasn't a dull moment.

The articles I've written to describe the installation of my PV system start next month. Let's just say that as far as commercially installed PV systems go, doing pretty much anything more than slapping a few PV panels on the roof and wiring an inverter or two next to your service panel is an invitation for adventure. Government PV subsidies go up and the voyage begins.

I'll introduce my installer in the article series, but I will say that I was lucky enough to partner with a team of people serious about getting the job done but not too proud to ask for help when they ran into obstacles. In the same vein, I think they appreciated that my objective in all this was to build and document a great PV system rather than take them to the cleaners over line-item responsibilities in a fixed-price contract. Surely they must have breathed sighs of relief when a test hole for the pole mounts hit a rock ledge and it wasn't a show-stopper. Rather than chuckle and say, "I guess this job just got a whole lot more expensive for you guys," I said: "Don't get excited. I know just the people who can fix this."

Of course, all of the adventure in installing a PV system comes from dealing with the problems and non-standard issues of actually generating your first watt. There are barely 200 residential PV installations in the whole state of Connecticut and most of those are roof-mounted arrays. After experiencing two types of PV installation methods, I can say that roof-installed panels are a snap and anything else is strictly experience-based knowledge. Certainly, my installer has a healthier appreciation of concrete and pole mounts than he had before dealing with me. I'm also sure the next time one of his salesmen says, "It just needs a few poles," he'll price the installation as if he were dynamiting it all out of solid rock. On the homeowner side, anyone installing pole mounts in the yard has to be prepared to remove a lot of trees and then landscape the whole place again after the cement trucks have destroyed everything. Yes, it was an adventure.

As I get a little more experience with the system and begin writing the third article, I'm confronted with yet another issue—remote monitoring. This also appears to be easier said than done.

Working with PV technology reminds me of the early days of computing. Back then, we thought nothing of building everything from scratch because all of us were engineers or trained professionals. If we needed a controller, we simply designed it. Today, ubiquitous and cost-effective computers mean we can purchase them like appliances and concentrate on the applications instead. The same lesson seems to apply to PV today. Most adopters think nothing of writing their own elaborate monitoring software or wiring probes and sensing elements to every panel. While I could conceivably do this too, I've done it enough times that I'm actually looking for excuses to avoid it.

My experience with PV installers thus far suggests that they know how to install net metering systems but haven't a clue what you are talking about when you say TCP/IP monitoring. That's why all of the PV-related articles you read contain so much homebrew hardware and software.

It's interesting to note that the Xantrex Technology designed inverters used in my PV system have the potential for extensive interactive monitoring, but at this point, Xantrex offers only a simple "local" LCD to present solar data, and there seems to be only a rudimentary user-supplied PC display program for the inverter's RS-232 data port. A pair of NETDIO serial-to-Ethernet modules configured for serial tunneling facilitated using the PC software on a single remote desktop, but it did nothing for web-based monitoring. That's when I started mumbling to myself.

Until I can really solve this problem the right way (and stop running up and down the stairs), I'll use the ultimate kludge. An hour ago, I went down to the garage where the inverters are mounted and screwed a power line Ethernet extender, a video camera, a video web server, and a 3-W LED light bulb to the wall over the Xantrex LCD. Yes, I know it's a total kludge (lose my name when you describe this technique to others), but at least I can call up a web page from anywhere and see how the system is functioning. Of course, if I were really insane, the next Rube Goldberg would be to attach a rubber mallet to a big web-controlled solenoid that whacks the front of the inverter to activate its vibration-sensitive display sequencing. I may wait on that one. ;-)

Fortunately, common sense will prevail. Xantrex tells me that they will have a web-monitoring system available soon. I'm also ordering some current transformers so that I can get a better real-time view of what the house is actually consuming and not just what I'm generating. Connecting everything will be an interesting project. I'll keep you informed. Of course, until then, let there be light.

steve.ciarcia@circuitcellar.com

PC Based Logic Analyzers

from \$499.00

NEW Model DV3400

- Faster Sampling (200/400 Msp/s)
- More Channels (36)
- More Memory (4x)
- Advanced Match circuits (8)
- $\pm 6V$ Adjustable Threshold (x2)
- Cascadable Sequencers (4)
- Enhanced Compression



Professional Hardware Capture + Software Analysis

- Automatic Real-time Hardware Compression eliminates the need to reduce resolution. Our newest version makes "dead time" insignificant.
- Edge and Pattern Triggers on all models. Advanced Model also includes Range ($>$, $<$, $=$, \leq , \geq , $<=$) and Stable Matches with Duration, and 4 flexible cascadable sequencers with pass-counters.
- Pattern Searches with Match & Duration.
- Specialized Sequential Searches for Serial and State Mode Signals.
- Display PC, Synchronous (SPI), Asynchronous (RS-232), State, Boolean, Bus and Analog Data.
- Single or Dual Waveform Views.
- Resolution Zoom in Wave Views.
- Time based Link Groups for all views.
- Specialized Exports from Data Tables and List Views.
- Multi-Signal Data Tables.
- Drag & Snap Markers.
- "Click to Center" function.
- "Snap Previous/Next" function.
- Print or Save Images with comments.
- USB 2.0 (480 Mbps).
- USB 1.1 compatible (12 Mbps).

Very flexible, "easy to configure" Advanced triggering.

PICmicro[®] MCU Programmer

Multi-Function, In-Circuit & Gang Operation

only \$199.00



TechTools

www.tech-tools.com

(972) 272-9392 • sales@tech-tools.com

Copyright © 2007 TechTools • DigiView, FlexROM, EconoROM and QuickWriter are trademarks of TechTools • PICmicro is a registered trademark of Microchip Technology Inc.

QuickWriter™

Do you **MoBo?**

Save \$20 on Parallax's **BASIC Stamp® 2pe Motherboard** (#28300; \$59.95) for a limited time. The "MoBo" provides a compact, professional-grade platform for BASIC Stamp applications. It includes a USB interface for programming, debugging and powering from an attached PC. With the MoBo and the growing assortment of plug-in daughterboards, you can integrate and package one-off or multiple application systems with ease.



BASIC Stamp 2pe MotherBoard
\$59.95



The **PWR-I/O-DB** (#28301; \$14.95) is a daughterboard that provides both power and a handy interface to potentiometers, servos, and Parallax's various three-pin sensor devices.



The **BS2PE Proto-DB Prototyping Daughterboard** (#28310; \$1.95) is a convenient through-hole board that allows you to build your own circuits with the MoBo motherboard.



The **DB-Expander Daughterboard-to-SIP Adapter** (#28325; \$9.95) provides the means to use daughterboards, designed for Parallax's motherboards, with solderless breadboards and other Parallax Products.



The **Daughterboard Extension Cable** (#800-28301; \$7.95) for convenient connection between the MoBo and daughterboards.



The **7Seg-DB Master Unit Daughterboard** (#28312; \$24.95) provide four digits of LED display, including alphabetic and punctuation capability. With the master module installed, up to seven additional **7Seg-DB Slave Unit Daughterboards** (#28313; \$24.95), may be added, daisy-chain-style, for a total of 32 contiguously-displayed digits.



The **TCS230-DB Color Sensor** (#28302; \$59.95) is a complete color detector, including a TAOS TCS230 RGB sensor chip, white LEDs, collimator lens, and standoffs to set the optimum sensing distance.



The **MoBo Power Cable** (#800-28300; \$3.20) allows you to connect external power to the MoBo without the need for the Power I/O daughterboard.

Order the **MoBo, Daughterboards and accessories** at www.parallax.com or call our Sales Department toll-free at 888-512-1024 (Mon-Fri, 7am-5pm, PDT).

PARALLAX 
www.parallax.com

BASIC Stamp is a registered trademark of Parallax Inc.
Parallax and the Parallax logo are trademarks of Parallax Inc.