PROJECT: A DIY PCB Preheating Setup
LOCATION: United States
PAGE: 36

INSIGHT: ADCs Explained
LOCATION: United States
PAGE: 50

HOW TO: Build a Robust Power Supply
LOCATION: Canada
PAGE: 56

CIRCUIT CELLAR

THE WORLD'S SOURCE f OR EMBEDDED ELECTRONICS ENGINEERING INf ORMATION

JANUARY 2012
ISSUE 258

EMBEDDED APPLICATIONS

A Scalable RTOS for Streamlining Embedded Development

A Look at X-Ray Tech & CMOS Imaging Sensors

Construct a Timeserver & Clock Synchronization System

Liquid Volume Measurement with an MCU & Simple Sensor

PLUS

Innovation in Research & Design

Lessons from an Experimental Engineer

// Hands-On MCU-Based Designs
// Novel Software & Graphics Projects
// Connecting Academia & Industry
// And More

7 25274 75349 9 01>
$7.50 U.S. ($8.50 Canada)
www.circuitcellar.com

The easiest finished product you will ever create

Start with the PK70 EX



Design your own, or use a NetBurner Personality Blade



Add your label to get a finished product



Personality Blades

Prototype Blade *(P/N NBPKBP-100CR)* ..... **$25 ea.**
Create your own Personality Blade

CPLD Digital I/O Blade *(P/N NBPKD-100CR)* ..... **$59 ea.**
Programmable Xilinx CPLD
32 digital I/O

Multi-I/O Blade *(P/N NBPKM-100CR)* ..... **$99 ea.**
Eight 12-bit analog-to-digital inputs
Two 16-bit digital-to-analog outputs
16 digital I/O

FPGA Digital I/O Blade *(P/N NBPKX500-100CR)* ..... **$169 ea.**
Programmable Xilinx Spartan 3E FPGA
High-density 62-pin connector

NTP Server with GPS Blade *(P/N NBPKX500-100CR)* ..... **$99 ea.**
Network Time Protocol (NTP) server
Precision GPS time reference

Quad UART Blade: RS-232 Version
(P/N NBPKBU-232CR) ..... **$99 ea.**
Network enable up to 4 RS-232 serial devices

Quad UART Blade: RS-422/485 Version
(P/N NBPKBU-485CR) ..... **$119 ea.**
Network enable up to 4 RS-422/485 serial devices

Quad UART Blade: RS-232/422/485 Version
(P/N NBPKBU-MMSCR) ..... **$245 ea.**
Network enable up to 4 RS-232/422/485 serial devices

PK70 EX & Development Kit

PK70 EX *(P/N PK70EX-100IR)* ..... **$199 ea.**
32-Bit Freescale ColdFire 147 MHz CPU
8MB SDRAM • 4MB Flash Memory
SD/MMC Flash Card Support • I2C • SPI • Address and Data Bus
10/100 Ethernet • 2 UARTs

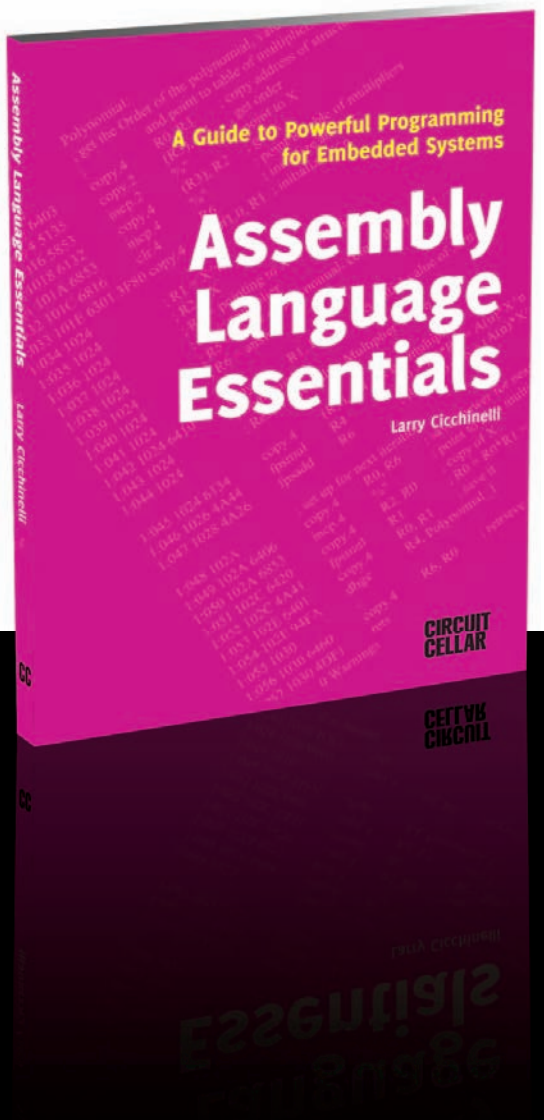
PK70 EX Development Kit
(P/N NNDK-PK70EX-KIT) ..... **$299 ea.**
The Development Kit features NetBurner's Eclipse, an enterprise level professional IDE that provides editing, downloading and debugging in one environment.

NetBurner
Networking in One Day!

Information and Sales | sales@netburner.com **Web** | www.netburner.com **Telephone** | 1-800-695-6828

Assembly Language Essentials

Circuit Cellar's first book, ***Assembly Language Essentials***, is a matter-of-fact guide to Assembly that will introduce you to the most fundamental programming language of a processor.



Author Larry Cicchinelli provides readers with:

- An introduction to Assembly language & its functionality
- Essential terminology pertaining to higher-level programming languages & computer architecture
- Important algorithms that may be built into high-level languages — multiplication, division, and polynomial evaluation
- Overview of Interrupt Service Routines
- Free, downloadable Assembler program …

and more!

On Sale NOW!

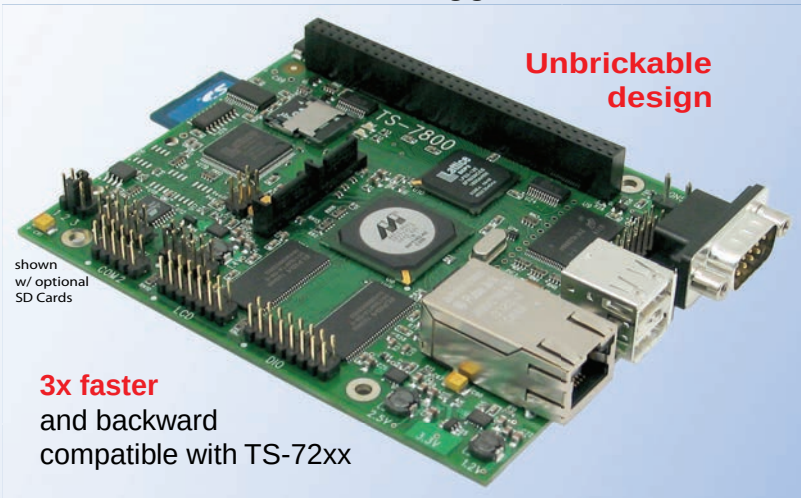
$47.50

WWW.CIRCUITCELLAR.COM • WWW.CC-WEBSHOP.COM

CIRCUIT CELLAR
THE WORLD'S SOURCE FOR EMBEDDED ELECTRONICS ENGINEERING INFORMATION

Embedded Systems

High-End Performance with Embedded Ruggedness

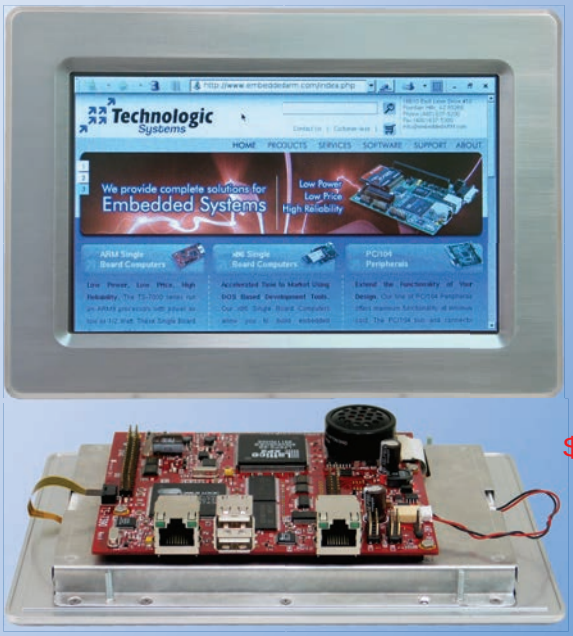


TS-7800
500 MHz ARM9

- Low power - 4W@5V — $229 qty 100
- 128MB DDR RAM
- 512MB high-speed (17MB/sec) onboard Flash — $269 qty 1
- 12K LUT customizable FPGA
- Internal PCI Bus, PC/104 connector
- 2 host USB 2.0 480 Mbps
- Gigabit ethernet
- 2 SD sockets
- 10 serial ports
- 110 GPIO
- 5 ADC (10-bit)
- 2 SATA ports
- Sleep mode uses 200 microamps
- Boots Linux 2.6 in 0.7 seconds
- Linux 2.6 and Debian by default

TS-TPC-7390
7" Color Touch Panel Computer

- Low Power, Industrial Quality Design
- Mountable aluminum frame
- 200 MHz ARM9
- 64MB SDRAM (128MB opt)
- 512MB Flash w/ Debian Linux
- 800 x 480 video core
- Dedicated framebuffer- 8MB RAM
- Audio codec with speaker
- Boots Linux 2.6 in less than 2 seconds
- Unbrickable, boots from SD or NAND
- Runs X Windows GUI applications
- Runs Eclipse IDE out-of-the-box



$449 qty 1

More Touch Panel Computers on our website

- Over 25 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

TS-WIFIBOX-2

A Complete Solution for 802.11g WiFi Applications

NEW!

$185 qty 1

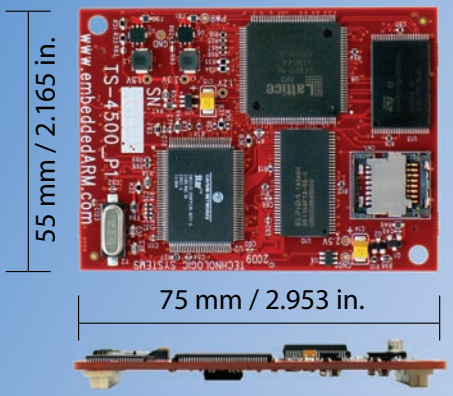


- Low power (3.2 watts), fanless
- Power via 5-12 VDC, USB, PoE (opt.)
- 64MB DDR-RAM
- 256MB ultra-reliable XNAND drive
- Micro-SD Card slot
- RS-232, RS-485, CAN, RTC
- Header with SPI and 11 DIO
- 480Mbit/s USB, Ethernet, PoE option
- Boots Linux 2.6.24 in < 3 seconds
- Un-brickable, boots from SD or flash
- Customizable FPGA - 5K LUT
- Optional DIN mountable enclosure

Ideal for gateway or firewall, protocol converter, web server, WiFi audio, and unattended remote applications

NEW!

series starts at $92 qty 100



- Dual 100-pin connectors
- Secure connection w/ mounting holes
- Common pin-out interface
- Low profile w/ 6mm spacing

TS-SOCKET Macrocontrollers

Jump Start Your Embedded System Design

TS-SOCKET Macrocontrollers are CPU core modules that securely connect to a baseboard using the TS-SOCKET connector standard. COTS baseboards are available or design a baseboard for a custom solution with drastically reduced design time and complexity. Start your embedded system around a TS-SOCKET Macrocontroller to reduce your overall project risk and accelerate time to market.

Current projects include:

- TS-4200: Atmel ARM9 with super low power
- TS-4300: Cavium ARM11 with dual 600 MHz and FPU
- TS-4500: Cavium ARM9 at very low cost
- TS-4700: Marvell PXA168 with video and 1.2 GHz CPU
- TS-4800: Freescale iMX515 with video and 800 MHz CPU
- Several COTS baseboards for evaluation & development

Technologic Systems

We use our stuff.

Visit our TS-7800 powered website at

www.embeddedARM.com

TASK MANAGER

Embedded Endeavors in 2012

What better way to begin the new year than by starting a project to enter in an international design competition? Go to www.chipkitchallenge.com to learn about the DesignSpark chipKIT Challenge. The idea is to use your engineering skills and ingenuity to create an innovative, energy-efficient design solution. Are you up for the challenge?

Sometimes the hardest part about designing and building an embedded application is getting started. Fortunately, you can use this edition of *Circuit Cellar* as a point of departure.

Turn to page 14 to learn how to build your own time-server system. David Penrose's design listens to a GPS, extracts the time and date, and retransmits this information to remote systems.

Are you planning a rapid prototyping application around an NXP Semiconductors mbed module? If so, check out the RTOS Andrew Levido designed for mbed (p. 24). His "mbos" is a true preemptive multitasking RTOS that's scalable to support virtually any project developed on the NXP Semiconductors LPC1768 platform.

There are many advantages associated with surface-mount devices (SMDs), but prototyping and repair work isn't always simple. Thus, Steve Johnson designed the "SMD Air Bath," which enables him to preheat a PCB assembly prior to the reflowing process with a hot air pencil (p. 36).

If you're a long-time *Circuit Cellar* reader, you're likely familiar with Mike Smith's work. He wrote more than 20 articles for *Circuit Cellar* since 1992. Flip to page 46 for an in-depth interview with Mike.

On page 50, columnist George Martin presents yet another set of lessons he's learned during his long career. This month he focuses on ADCs and their supporting hardware and software.

Part of being a consummate engineer involves understanding power supplies and how to maximize system power. On page 56, George Novacek starts an article series on power supply design. The article in this issue covers switching regulator modules, power dissipation, and low-voltage operation.

Embedded technology is required to analyze the bones embedded in the human body. Columnist Richard Wotiz demystifies X-ray technology with an interesting article about CMOS imaging and modern digital sensors (p. 60).

The New England region of the U.S. experienced some wild weather during the last quarter of 2011: Hurricane Irene in late August, a crazy October storm that dropped more than 18" of snow in some places, and some uncommonly warm days in November. Those events were likely the impetus for Jeff Bachiochi's article "Quantifying Precipitation." Turn to page 68 to learn how to build a DIY, MCU-based rainfall measurement system.

Good luck with your 2012 projects!

cj@circuitcellar.com

C. Abate

CIRCUIT CELLAR®

THE WORLD'S SOURCE FOR EMBEDDED ELECTRONICS ENGINEERING INFORMATION

FOUNDER/EDITORIAL DIRECTOR
Steve Ciarcia

EDITOR-IN-CHIEF
C. J. Abate

ASSOCIATE EDITOR
Nan Price

CONTRIBUTING EDITORS
Jeff Bachiochi
Bob Japenga
Robert Lacoste
George Martin
Ed Nisley
George Novacek
Richard Wotiz

PROJECT EDITORS
Ken Davidson
David Tweed

PUBLISHER
Hugo Van haecke

ASSOCIATE PUBLISHER
Shannon Barraclough

ART DIRECTOR
KC Prescott

CONTROLLER
Jeff Yanco

CUSTOMER SERVICE
Debbie Lavoie

ADMINISTRATIVE COORDINATOR
Valerie Luster

MARKETING REPRESENTATIVE
Amanda Anderson

GRAPHIC DESIGNER
Grace Chen

Circuit Cellar's mission is to collect, select, and disseminate need-to-know information around the world in the fields of embedded hardware, embedded software, and computer applications. Circuit Cellar uses an assortment of print and electronic content-delivery platforms to reach a diverse international readership of professionals, academics, and electronics specialists who work with embedded, MCU-related technologies on a regular basis. Our aim is to help each reader become a well-rounded, multidisciplinary practitioner who can confidently bring innovative, cutting-edge engineering ideas to bear on any number of relevant tasks, problems, and technologies.

ADVERTISING

800.454.3741 • 978.281.7708 • www.circuitcellar.com/advertise

ADVERTISING REPRESENTATIVE
Peter Wostrel
Strategic Media Marketing, Inc.
1187 Washington St., Gloucester, MA 01930 USA
800.454.3741 • 978.281.7708
peter@smmarketing.us • www.smmarketing.us
Fax: 978.281.7706

ADVERTISING COORDINATOR
Kim Hopkins
E-mail: khopkins@circuitcellar.com

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

PRINTED IN THE UNITED STATES

CONTACTS

SUBSCRIPTIONS
Information: www.cc-access.com, E-mail: subscribe@circuitcellar.com
Subscribe: 800.269.6301, www.cc-access.com, 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® (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 $45, Canada/Mexico $60, all other countries $63. Two-year (24 issues) subscription rate USA and possessions $80, Canada/Mexico $110, all other countries $116.** 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.

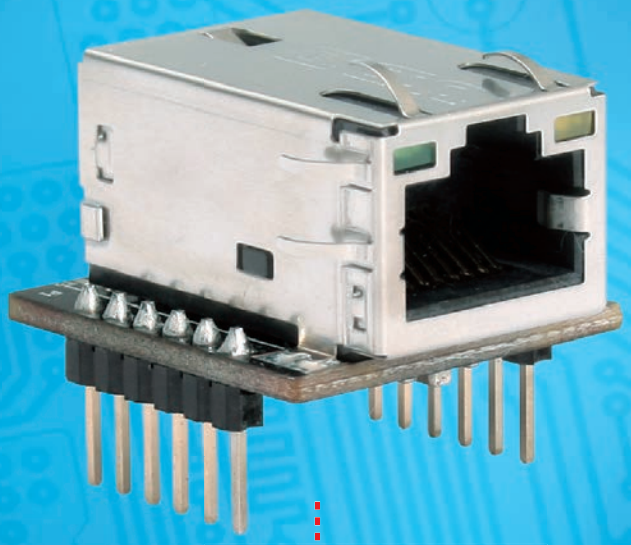
Circuit Cellar® makes no warranties and assumes no responsibility or liability of any kind for errors in these programs or schematics or for the consequences of any such errors. Furthermore, because of possible variation in the quality and condition of materials and workmanship of reader-assembled projects, 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 © 2011 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.

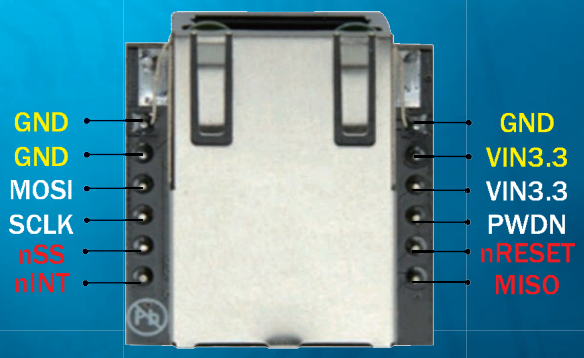
WIZnet

Embedded Internet is Just One Plug Away



WIZ820io Internet Offload Module

- Plug-in module with TCP/IP, 10/100 Ethernet MAC/PHY, transformer & RJ-45
- Breadboard Friendly
- 8 high-speed hardware sockets
- High-speed SPI Interface
- Power Down Mode & Wake-on LAN
- 23mm x 25mm PCB size
- Arduino Compatible



Get more information at www.WiznetTechnology.com
3003 N First Street, San Jose, CA 95134, USA TEL : 408-232-5415
E-mail : sales@wiznettechnology.com

INSIDE ISSUE 258

January 2012 • Embedded Applications

14 **A Different Timeshare**
Build a Timeserver System
Dave Penrose

24 **mbos**
A Real-Time Operating System for mbed
Andrew Levido

36 **The SMD Air Bath**
A DIY PCB Preheating Setup
Steve Johnson

Timeserver Design p. 14

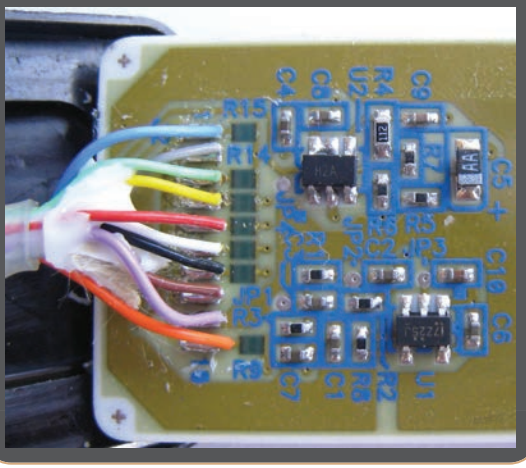


PCB Preheating System p. 36





Inside a Sensor Assembly p. 60



DesignSpark chipKIT™ Challenge

The DesignSpark chipKIT Challenge is underway. Use Microchip Technology's chipKIT Max32 development board to design an energy-efficient and eco-friendly device. Maximize your design power while reducing your environmental and energy footprint! Go to www.chipkitchallenge.com for more information.

50 **LESSONS FROM THE TRENCHES**
ADCs and Signal Filtering
George Martin

56 **THE CONSUMMATE ENGINEER**
Build a Robust Power Supply (Part 1)
Power Dissipation
George Novacek

60 **EMBEDDED UNVEILED**
Digital X-Ray Sensors
Richard Wotiz

68 **FROM THE BENCH**
Quantifying Precipitation
Construct a Device to Measure Rainfall
Jeff Bachiochi

TASK MANAGER 4
Embedded Endeavors in 2012
C. J. Abate

NEW PRODUCT NEWS 8

TEST YOUR EQ 13

QUESTIONS & ANSWERS 46
Innovative Research and Design
An Interview with Mike Smith

CROSSWORD 74

INDEX OF ADVERTISERS 79
February Preview

PRIORITY INTERRUPT 80
Too Many Layers
Steve Ciarcia

AS9120A
Authorized Distributor

mouser.com

Find It Here. Faster.™

The Newest Products for Your Newest Designs®

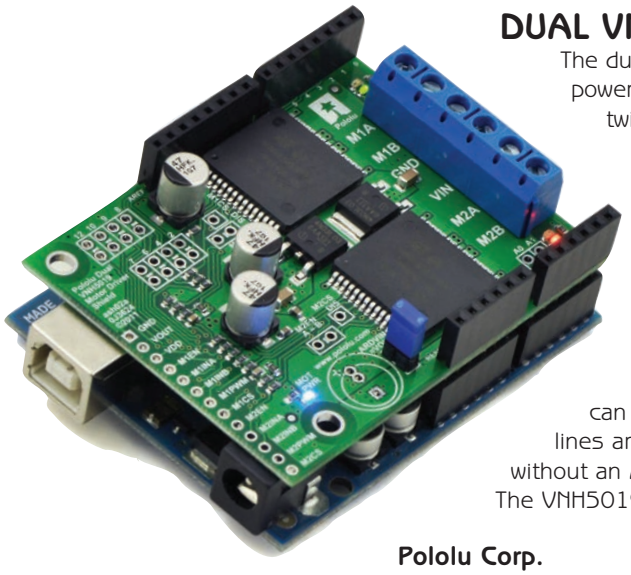
Scan Here
to Watch Video

Authorized distributor for the most advanced
semiconductors and electronic components.
Get What's Next. Right now at mouser.com.

MOUSER ELECTRONICS®
a tti company

Mouser and Mouser Electronics are registered trademarks of Mouser Electronics, Inc.

DUAL VNH5019 MOTOR DRIVER SHIELD FOR ARDUINO



The dual **VNH5019** motor driver shield for Arduino can control up to two high-power DC motors with an Arduino or Arduino-compatible device. The shield's twin robust VNH5019 motor drivers operate over a range of 5.5 V to 24 V. Each motor driver delivers a continuous 12 A (30-A peak) to an independently controlled motor, or the driver outputs can be combined to deliver a continuous 24 A (60-A peak) to a single motor. In addition, the motor drivers offer current-sense feedback and accept ultrasonic pulse-width modulation (PWM) frequencies for quiet operation.

The VNH5019 motor driver is intended for users ranging from beginners who prefer a plug-and-play motor-control solution for their Arduinos (and don't mind a little soldering) to experts who want to directly interface with ST's motor driver ICs. All the Arduino pin mappings can be customized if the defaults are not convenient. The VNH5019 control lines are broken out along the left side of the board for general-purpose use without an Arduino.

The VNH5019 motor driver shield costs **$59.95**.

Pololu Corp.
www.pololu.com

COMPACT FOUR-QUADRANT PWM SERVO CONTROLLER

The **ESCON 36/2 DC** is a four-quadrant pulse-width modulation (PWM) servo controller for use with DC motors up to 72 W. The digital current controller features a large bandwidth for optimal motor current/torque control. The drift-free, dynamic-speed behavior enables a speed range of 1 to 150,000 RPM. The controller provides a wide range of functions, with fully configurable digital and analog inputs and outputs. When combined with Maxon's range of motors, high demanding and dynamic drive solutions are possible. Additionally, the ESCON 36/2 DC can be run in various operating modes: speed controller (closed loop), speed controller (open loop), or current controller.



This ultra-compact servo controller is controlled with an analog set value. The value can be specified with analog voltage, an external or internal potentiometer, a defined value, or with a PWM signal with variable duty cycle. Other functions include the ability to enable or disable the power stage depending on the direction of rotation, or to use speed ramps for acceleration and deceleration. The speed can be regulated with a digital incremental encoder (two-channel, with or without line driver), a DC tacho, or without an encoder (IxR compensation).

The user-friendly controller is easy to start up. No in-depth knowledge of drive technology is required. When the servo controller is connected to a PC via a USB port, it can easily and efficiently be configured with the "ESCON Studio" graphical user interface (GUI). During startup and configuration of the inputs and outputs, monitoring, data recording, and diagnostics, the user has access to a variety of functions and is assisted by user-friendly software wizards, as well as an automatic procedure for fine-tuning the controller. The controller comes fully equipped with everything that is needed. No additional external filters or motor chokes are required, and preassembled cables for the start-up are available as accessories.

The ESCON 36/2 DC has protective circuits against overcurrent, excess temperature, under- and over-voltage, voltage transients, and short circuits in the motor cable. It is equipped with protected digital inputs and outputs and an adjustable current limitation for protecting the motor and the load. The motor current and the actual speed of the motor shaft can be monitored by means of the analog output voltage.

The large range for the input voltage and the operating temperature enables flexible use in a variety of drive applications. Contact Maxon for pricing.

Maxon Precision Motors
www.maxonmotorusa.com

NEW PRODUCT NEWS

DESIGNSPARK

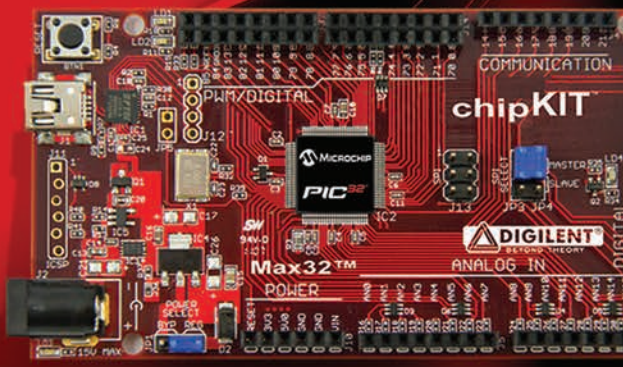
Turn a hot idea into a cool solution.

DesignSpark chipKIT™ Challenge

The DesignSpark chipKIT™ Challenge is well under way! Have you registered at **www.chipkitchallenge.com** yet? Hurry over and get involved in the DesignSpark community today. By submitting your proposal for an energy-efficient design solution, you will automatically be considered for a **chipKIT™ Community Choice Award***. At the end of January, one lucky participant will be rewarded for having the most creative project proposal.

Awarded by the DesignSpark team, the winner of this sub-challenge will receive a **$100 voucher for RS Components/Allie Electronics** and a **free digital subscription to *Circuit Cellar* and *Elektor* magazines!**

Register your project today!
Visit **www.chipkitchallenge.com** to participate.



Visit www.chipkitchallenge.com
for complete rules and details.

* Participation in the Community Choice Awards does not increase your chances of winning the Grand Prize with your Final Project(s) submission. The deadline for Final Project submissions is March 27, 2012. See website for more information.

IN ASSOCIATION WITH:

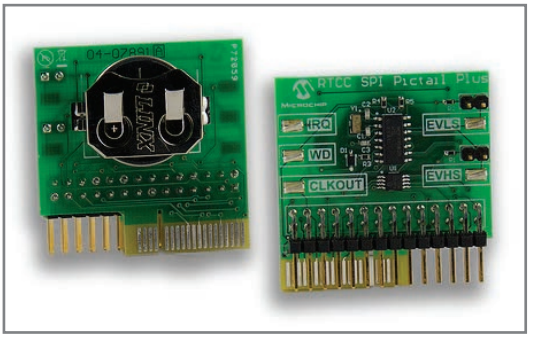
elektor · MICROCHIP · RS · CIRCUIT CELLAR · ALLIED ELECTRONICS · DIGILENT BEYOND THEORY

DesignSpark chipKIT™ Challenge

chipKIT™ is a registered trademark of Microchip Technology Inc. Max32™ is a registered trademark of Digilent, Inc.

BATTERY-BACKED RTCC DEVICES WITH 10-MHz SPI & WATCHDOG TIMER

The **MCP795W*xx*/B*xx*** family of standalone real-time clock/calendar (RTCC) devices features a 10-MHz SPI and nonvolatile memory. The RTCC devices reduce component counts and costs, making them well suited for a variety of industries: smart-energy (e.g., thermostats, power meters, and commercial refrigeration); home appliance (e.g., coffee makers, stoves, and microwave ovens); automotive (e.g., dashboard controls, car radios, and GPS); consumer electronic (e.g., office equipment and video systems); and communications (e.g., radios, cordless phones, and network systems).



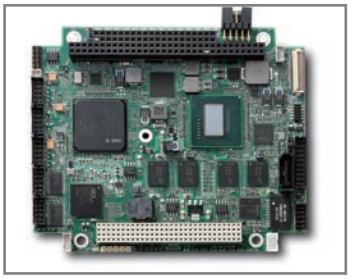
The MCP795W*xx*/B*xx* RTCC devices feature millisecond alarms. Their 10-MHz SPI supports fast data access providing longer microcontroller Sleep and Power-down modes and less power draw from the battery. Other low-power features include two event-detect inputs that draw minimal current in the RTCC device but wake the microcontroller when events such as a switch press or system tampering have occurred. The devices also include three types of nonvolatile memory—EEPROM, SRAM, and a Unique ID—which can be ordered blank or with a preprogrammed MAC address as a standard product to further reduce component counts and lower costs associated with serialized programming. Along with the automatic battery switchover for back-up power, a power-fail time stamp logs the time and duration of any power failure. For accurate timekeeping over a broad temperature range, the devices feature a wide digital-trimming range that compensates for crystal frequency drift over temperature. In designs that require a low-frequency clock source, a boot clock option in the RTCC can provide a 32-kHz clock output at power-up and eliminate the need for a second crystal.

Pricing for the MCP795W*xx* and MCP795B*xx* RTCC devices starts at **$1.18** each in 10,000-unit quantities.

Microchip Technology, Inc.
www.microchip.com

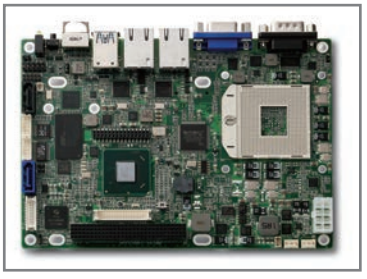
SBCs WITH INTEL ATOM E600 & INTEL CORE I7 PROCESSORS

ADLINK has introduced two new single-board computers (SBCs): the Ampro **Extreme Rugged CoreModule 720** PC/104-Plus SBC with Intel Atom Processor E600 series and **Rugged ReadyBoard 910** EPIC SBC with second-generation Intel Core i7/i5/i3 and Celeron processors. Both SBCs feature on-board solid-state drives (SSDs) and provide a compact form factor designed for applications in harsh environments including transportation, self-service, digital signage, and video surveillance.



Extreme Rugged CoreModule 720

The CoreModule 720 is based on the Intel Atom Processor E600 series from 600 MHz up to 1.6 GHz. The PC/104-Plus stackable form factor enables customers to build low-power solutions for space-constrained, extremely rugged environments. The CoreModule 720 is a PC/104-Plus SBC with PCI and ISA bus connectivity. It provides an integrated 4-GB SSD, CAN bus, SATA, and a range of peripheral I/O support. Additional features include support for up to 2 GB soldered DDR2 SDRAM at 600/800 MHz and 24-bit LVDS and SDVO graphics. The Intel Platform Controller Hub EG20T accommodates a wide range of common I/Os, such as USB, SATA, GbE, SDIO, Serial, and CAN bus. Designed to meet demanding shock and vibration requirements, the CoreModule 720 uses 50% thicker printed circuit board (PCB) and supports an extended temperature range of –40°C to 85°C.



Rugged ReadyBoard 910

The Ampro by ADLINK ReadyBoard 910 integrates the 2nd Generation Intel Core i7/i5/i3 and Celeron processors (Socket G2) and mobile Intel HM65 Express chipset, onboard SSD, and robust I/O in a compact EPIC form factor. The ReadyBoard 910 supports three display interfaces, including analog VGA, LVDS, and DVI-D. It features dual gigabit Ethernet, SuperSpeed USB 3.0 with 5 Gb/s data transfer rate, PCI Express Mini Card socket, and PCI-104 expansion.

The Extreme Rugged boards and systems are designed for harsh environments. Robust test methods—including highly accelerated life testing (HALT)—ensure optimal product design phases and meet stringent requirements, including –40°C to 85°C operating temperature range, MIL-STD, shock and vibration, and long-term reliability.

Contact ADLINK for pricing.

ADLINK Technology, Inc.
www.adlink.com

LED DRIVER WITH SERIAL INTERFACE FOR EASY PROGRAMMING

The **NCP1840** is a multi-LED driver that can drive up to eight LEDs independently using any user-created illumination pattern. The driver is designed for multi-LED status indicators that communicate status and diagnostic conditions to the user, and can help differentiate an end product via LED animation.

Engineers can use the NCP1840 to offload some of the work normally handled by the processor in their system design, as the processor no longer needs to directly control each individual LED. Current- and dimming-level programming is conveniently done by writing to and reading registers via a serial interface that uniformly controls the brightness and blink rate of each LED. LED brightness levels can be logarithmically controlled—as opposed to linearly controlled—to accommodate natural eye viewing with no brightness distortion.

The NCP1840 can extend battery life by using a high-efficiency, low-noise, quad-mode charge pump, which makes it well suited for portable applications. The charge pump and constant current regulators maintain forward LED voltage and active operation despite swings in input voltage. The NCP1840 also saves on space by requiring only five small ceramic capacitors and one resistor to form a complete LED power supply and current controller.

Ideal for applications in the industrial, computing, and consumer end markets, the device is offered in a Pb-free, RoHS-compliant 4 mm × 4 mm QFN-20 package, and supports an operating temperature range of –40°C to 85°C.

The NCP1840 costs **$0.94** per unit in 1,000-unit quantities.



ON Semiconductor
www.onsemi.com

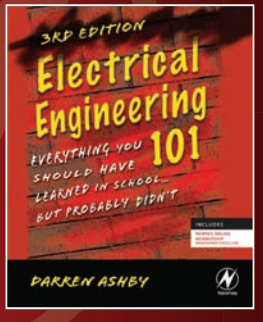
NPN

Newnes Press

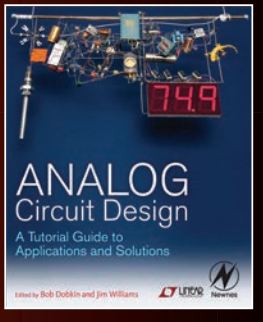
Education Never Ends. Everything you need to know to get started.



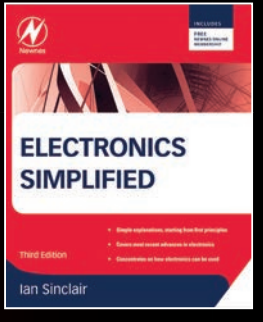
By: Keith Brindley
ISBN: 9780080969923



By: Darren Ashby
ISBN: 9780123860019



By: Bob Dobkin and Jim Williams
ISBN: 9780123851857



By: Ian Sinclair
ISBN: 9780080970639

Newnes

Save 20% on all titles when you purchase from www.newnespress.com.
Enter promotional code **Newnes30** at checkout.

Find us on Facebook!

Register for our e-news at newnespress.com
Receive our best discounts · Hear about books before they publish
Access to free sample chapters, video tutorials and more!

by Ian Bromley

DesignSpark chipKIT™ Challenge

It's time for a new design challenge to showcase the functionality of the chipKIT™ Max32™ development kit and the award-winning DesignSpark PCB software tool in environmentally friendly applications.

Towards the end of 2011, the world's population passed the seven billion mark and the U.N. estimates are that it could be closer to nine billion within a few decades. Add to that the increasing energy demands to meet growth required by major emerging economies, and there is no doubt that meeting the world's energy needs while also looking after our environment is one of the great challenges of our time. In the drive to achieve greater energy sustainability, clearly the development of innovative new solutions are required, and I believe embedded electronics can make a contribution to meet this global challenge.

It isn't just about developing ultra low power electronic devices—in fact, arguably more important is increasing the energy efficiency of electronic systems. Maximizing power output and delivering the required performance while reducing the environmental and energy footprint is becoming a highly significant ratio. For example, in the embedded world, a 32-bit microcontroller is most likely to consume more power than an 8-bit device, but it also has the ability to perform very fast calculations and process algorithms that can significantly increase energy efficiency in motor-control applications. So, when we were approached last year by Elektor and Circuit Cellar to participate in a worldwide design competition, the choice was an easy one: it had to be about energy efficiency.

Officially launched in late November at Elektor Live! and offering total cash prizes of $10,000, including a first prize of $5,000, the DesignSpark chipKIT™ Challenge is all about encouraging engineers, students, and hobbyists to think about the use of power and develop solutions that will increase energy efficiency while also maintaining an eco-friendly footprint. The prize-winning application can be anything that reduces the energy footprint of a system. For example, it could be an energy-efficient battery charger, a controller for a windmill, or perhaps an energy-usage management device for use in the home.

Competition entrants are now developing energy-efficient and environmentally friendly applications based on the chipKIT™ Max32™ development platform from Digilent, which features Microchip's 32-bit PIC32 microcontroller. The first 1,000 registered entrants have already received a complimentary chipKIT™ Max32™ development board.

The chipKIT™ Max32™ development platform is a 32-bit Arduino-compatible solution that enables engineers, students, and hobbyists to easily and inexpensively integrate electronics into their projects. The chipKIT™ hardware is compatible with existing 3.3-V Arduino shields and applications, and can be developed using a modified version of the Arduino IDE and existing Arduino resources, such as code examples, libraries, references, and tutorials. The chipKIT™ Basic I/O Shield is compatible with the chipKIT™ Max32™ board and offers users simple push buttons, switches, LEDs, I²C EEPROM, I²C temperature sensor, and a 128 × 32 pixel organic-LED graphic display.

All entries must include an extension card developed using RS's free-of-charge DesignSpark PCB software tool with code compiled using Microchip's MPLAB® IDE software. The DesignSpark PCB software tool is unique in the industry, and since its launch in July 2010, more than 100,000 users have downloaded DesignSpark PCB, RS's professional-standard PCB design software, which has proven to be one of the most popular free PCB design software packages available. Thousands of users have contributed suggestions for its continued development via the DesignSpark community, which is available at www.designspark.com.

During the competition, which finishes at the end of March, entrants are being strongly encouraged to engage and interact with other members of the online DesignSpark community by posting information on their projects, providing updates on progress, and sharing comments and ideas on their respective designs. Participants will automatically qualify for entry into bonus Community Choice Awards, in addition to admission into spot prize draws for the best collaboration to win vouchers exchangeable for products ordered from RS Components/Allied Electronics.

The competition entries will be judged on the level of energy efficiency and the quality of the extension card's PCB design. Entries are due on March 28, 2012 and the winners will be announced in April 2012.

Further details and registration for the DesignSpark chipKIT™ challenge are available at: chipkitchallenge.com.

Ian Bromley is a Technical Marketing Engineer at RS Components and the Project Manager for the DesignSpark PCB software tool. Prior to working for RS, Ian worked for many years as a design support consultant with Texas Instruments, in addition to working as a field applications engineer immediately following his graduation in 1994 with an honours degree in Microelectronic Engineering.



To register and enter the DesignSpark chipKIT™ Challenge, go to: www.chipkitchallenge.com

CIRCUIT CELLAR

Test Your EQ

A need came up recently in conjunction with the restoration of an antique electronic computer that uses some odd logic levels for a circuit that would translate from 0 V/5 V logic to 0 V/–5 V logic. In other words, when the input is at 0 V, the output should be at 0 V, and when the input is at 5 V, the output should be at –5 V. The circuit in Figure 1 was proposed. Let's dig into its operation in detail.

Problem 1—What is the function of the Q1-Q2 transistor pair, and what is the significance of the node labeled "VTH"?

Problem 2—What are the functions of resistors R1 and R2?

Problem 3—What are the input characteristics of the circuit? In other words, what sort of load does the driving circuit see?

Problem 4—The circuit in Figure 2 was proposed as an alternate. In what ways is it better? In what ways is it worse?

Contributed by David Tweed

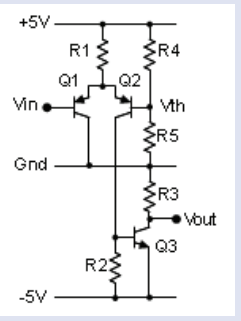


Figure 1

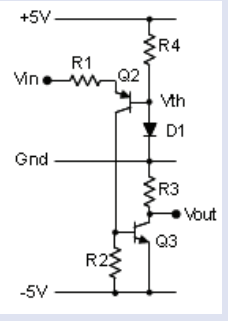


Figure 2

What's your EQ?—The answers are posted at www.circuitcellar.com/eq/

You may contact the quizmasters at eq@circuitcellar.com





by Dave Penrose (USA)

A Different Timeshare

Build a Timeserver System

You can build a timeserver that listens to a GPS signal, extracts the time and date, and retransmits this information to remote systems. The design manages more than one microcontroller that has a real-time clock with an I²C interface and keeps all the clocks synchronized.

In this article, I'll describe how I designed and built my Timeshare system (see Photo 1). A precise and reliable time word is derived from a global positioning system (GPS) receiver, converted to local time, and broadcast locally to synchronize multiple real-time clocks. Inexpensive transmitters and receivers extend the availability of the GPS time to areas where the GPS signal is unreadable. A pre-emphasis process is used to improve the readability of the local time transmission and a multimaster inter-integrated circuit (I²C) bus is implemented to ensure error-free transfer to the target clock.

WWV & WWVB BROADCASTING

A number of sources for accurate time information are available including WWV and WWVB broadcasting on shortwave and low frequency from Fort Collins, CO. Internet time information, cable TV time information, caller ID time information, and GPS time data from the GPS satellite array. Each of these sources has advantages and disadvantages based on system connectivity and distance from the time source. The low-frequency WWVB signal is one of the easiest and most inexpensive to exploit, but the signal strength in regions remote from Colorado makes reception difficult, and processing the signal usually requires significant error correction to ensure reliability. Internet data, as well as telephone and cable TV signals, requires connection to these networks. The GPS signal is almost universally available as long as the receiver has a reasonably good view of the sky.

For my design, I decided to use the GPS signal and then process and retransmit the time using a low-cost On Off Keyed (OOK) transmitter. My decision was based on the characteristics of these various data and the desire to implement a simple system that wasn't connected to a network. Although OOK devices suffer from less reliable

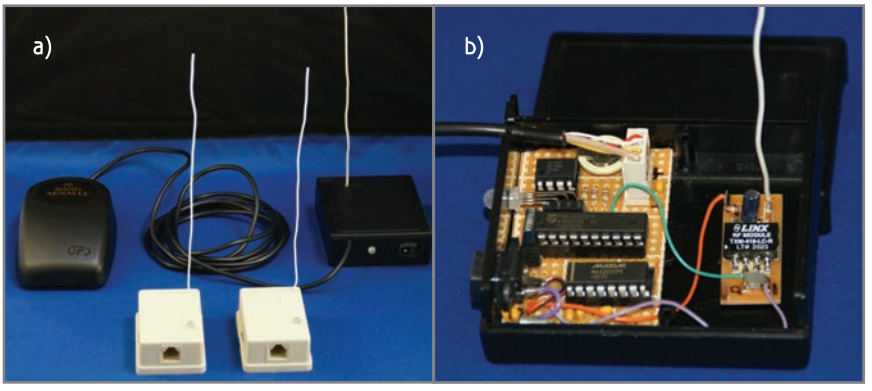


Photo 1a—You can see the Timeshare master transmitter connected to its GPS receiver. Two Timeshare slave receivers are in the center. The transmitter has a power jack for 5 V and a red/green/blue LED on the front of its case. The receivers have an RJ-45 jack for the I²C and power connection and a red/green/blue LED on the top of the cases. **b**—This internal view of the Timeshare transmitter shows the two circuit boards for the project. The digital board has an NXP 89LPC925 microprocessor in the center surrounded by the real-time clock in the upper part of the board and the serial level converter chip in the lower section. Note that capacitors are present to condition the power to the Linx module. A resistor is included to limit voltage. These may not be required depending on your configuration.

communications than FM devices, I decided to use the OOK devices for their small size and low cost and then add software to improve the probability of correct reception. This retransmission provided time to locations that could not receive the GPS signal. Using the GPS signal meant that system checkout could occur at any time, not just at night when the WWVB signal was best. Unfortunately, it also meant giving up the indicator of daylight saving time that is included in the WWVB signal, but this data could be easily inserted into the retransmitted data based on some simple processing.

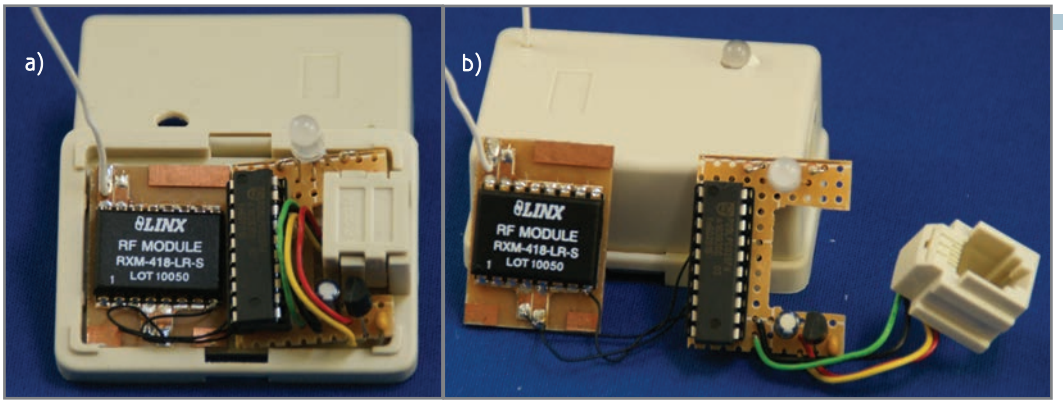


Photo 2a—The receiver portion of the Timeshare system is packaged in a surface-mount RJ-45 box that I carved up to eliminate the internal screw connectors. What is left is enough room for the digital and RF circuit boards and the RJ-45 connector. b—Separating the boards that make up the Timeshare slave receiver shows how every available bit of real estate was used to implement the receiver. The red/green/blue LED is on the upper portion of the digital board and the power conditioning is on the lower portion of the board. You can't see that the power decoupling capacitor is placed between power and ground inside the DIP socket for an NXP 89LPC925. This saves space and also provides great performance.

Processing the GPS signal as output from commercial GPS receivers is simple since the data is incorporated into standard sentences defined by National Marine Electronics Association (NMEA). NMEA-defined data is a standard 4,800-bps serial datastream that incorporates time, position, and other data from the GPS.

Having decided on a GPS source limited the number of places the receiver could be located to still achieve a reliable lock on the satellites. An upstairs location close to a window generally provides good reception. Other areas in a building generally prevent reception, but by retransmitting the time on 418 MHz, the receiver can reach a significant distance both inside and outside a structure. The retransmitted data will have noise, which means that for reliability there needs to be some minor processing at the receiving end to extract the time words. This processing at the receiver and the extraction and rebroadcast of the time from the GPS is easily accomplished with low-cost microprocessors. The transmitters and receivers are also widely available and low in cost. This project used Linx Technologies's TXM/RXM series, which operates on 418 MHz.

HARDWARE

Almost all commercial GPS devices output the processed GPS data in NEMA sentences. A number of low-cost devices that omit the display and rely on a laptop computer for processing are available. I selected an older Rand McNally device that uses a RS-232 output and 5-V power for this system. Newer devices in this family use a USB interface so they are more difficult to adapt for a microprocessor interface. There are also a large number of GPS receiver circuits available that provide the proper output and are low in cost. These provide a second, very attractive approach to receiving the GPS time information.

The processor selected to process the GPS data was an NXP Semiconductors 89LPC925, which is a small, 8-bit, 3-V system that is 5-V tolerant. It is easily programmed and has a large family of software available for its core 8051-type system. A battery backed-up Maxim Integrated Products DS1307 real-time clock (RTC) is included in this master system to enable the system to flywheel through periods when the GPS receiver loses lock on the satellite constellation.

As I mentioned earlier, the Linx Technologies TXM-418-LC transmitter module retransmits the time data. Photo 1a shows the completed master transmitter together with two of the slave devices. Photo 1b shows an internal view of the transmitter with the digital processing equipment and the transmitter mounted to separate PC boards.

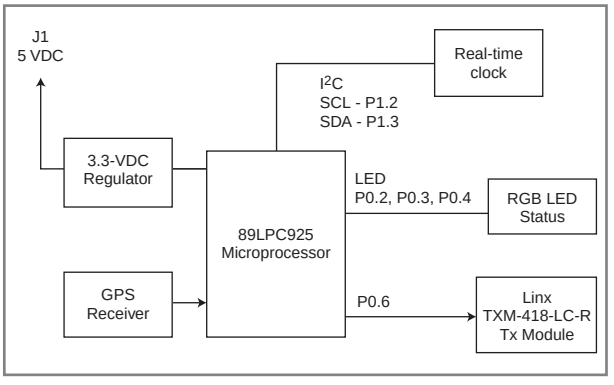


Figure 1—The Timeshare GPS time retransmitter shows the relationships between the GPS receiver, the microprocessor, and the Linx transmitter. Precise time is extracted from the GPS datastream by the microprocessor and retransmitted on 418 MHz by the Linx transmitter. The status on the GPS is indicated on a red/green/blue LED.

I assembled the slave receiver from a Linx Technologies RXM-418-LR receiver chip and a second NXP 89LPC925 microprocessor. Some complexity is involved in passing the time word to the target since both the receiver and the target processor need to be masters for the I²C bus. The I²C bus specification enables this through a process called "multimaster." The 89LPC925

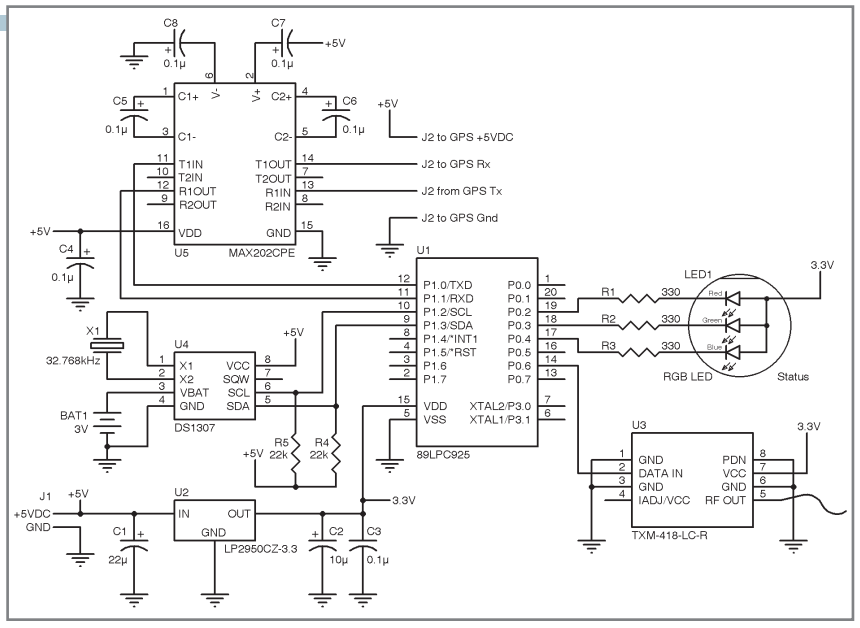


Figure 2—The heart of the Timeshare system is an NXP 89LPC925 microprocessor. It reads the serial bitstream from the GPS receiver through a Maxim MAX202 level converter chip transceiver. The current time is extracted from this datastream and stored in a Maxim DS1307 RTC. Every four minutes this time is read from the clock chip and transmitted to remote slaves using the Linx transmitter module.

processor has hardware for the I²C bus and includes processing that enables this multimaster configuration to be implemented. If this hardware assist from the 89LPC925 had not been available, then the bus communications could have been implemented with a software approach. But it becomes more difficult since the software needs to constantly track the state of the bus and monitor each bit transmitted for collision. Internal views of the completed slave unit with the microprocessor and receiver on separate PC boards are shown in Photo 2.

IMPLEMENTATION

Implementation of the project was divided into a number of discrete steps. The master transmitter unit was implemented first since it was more complex than the receiver. This implementation included hardware interface to the GPS receiver, software decoding of the NMEA sentences, local time from universal coordinated time (UTC) conversion, daylight saving time logic, flywheel clock to bridge GPS outages, software interface to the Linx transmitter, and hardware interface to the Linx transmitter. A block diagram of the transmitter is shown in Figure 1. The schematic is shown in Figure 2.

The slave receiver was much simpler with the following tasks: hardware interface to the Linx receiver, software interface to the receiver, and I²C interface to the external clock. Figure 3 is a block diagram of the receiver. The schematic is shown in Figure 4. Each of these steps will be described. Even if you are not interested in reconstructing the entire project, some pieces of this may be useful for you. One note of interest is, if you examine the code on both the master and the slave, the master uses what are called "fat" interrupts, where most of the processing takes place in the interrupt. The slave is a more typical "skinny" interrupt structure, where the processing takes place outside of the interrupt. The master logic, although it looks complex, has only a few steps that are executed in the interrupt handler, so this solution seems to work fine.

MASTER: INTERFACING TO GPS DEVICE

The output from most GPS devices is a 4,800-bps standard RS-232 serial bitstream. Most devices also offer optional output formats so you need to check your system set-up menu for the device to make sure NMEA is selected. These menus also enable you to vary the output baud rate, but the default 4,800 worked quite well for this project. To read the RS-232 signal, the levels need to be converted to voltages appropriate for the processor. The NXP 89LPC925 is 5-V tolerant, so I chose the simple solution of a Maxim Integrated Products MAX202 transceiver. This is overkill since only the receive side and only one of the two available channels are used.

Decoding of NMEA output was designed to search for the GPS required minimum specific GPS/transit data (GPRMC) sentence. Once found, the validity flag, time, and date information were extracted. A printout of the datastream output by a GPS device just prior to it achieving lock and just following lock is shown in Figure 5. For this figure, blank lines have been added between the blocks of data and the location information has been modified.

Looking through the data you will notice that, prior to lock, the GPRMC data is quite short with a number of commas indicating missing information. These markers are always output, so scanning this data for the three fields is easy. During this search and decode process the project indicates its status by means of a red/green/blue

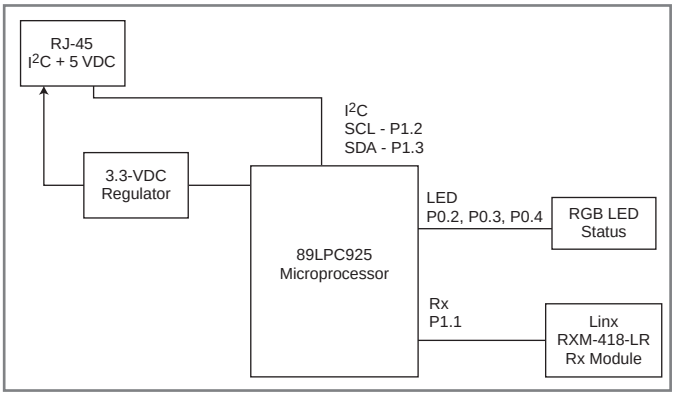


Figure 3—The Timeshare remote receiver is simple. The precise GPS time information is received by the Linx receiver module, error checked by the microprocessor, and written to a Maxim DS1307 RTC in the target over the I²C interface.

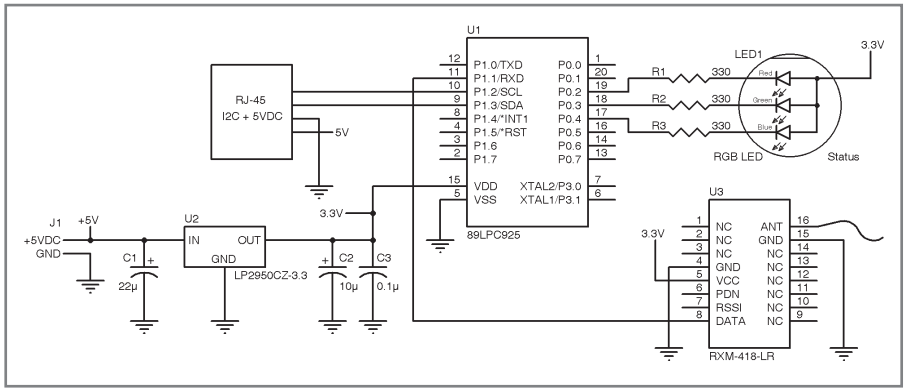


Figure 4—A standard interface for the I²C bus in the target projects consists of RJ-45 connectors. This remote receiver plugs into one of the available RJ-45 jacks on the target and updates the target's Maxim DS1307 RTC with the time from the GPS transmitter. The status of the link between the transmitter and this receiver is indicated on the multicolored LED connected to an NXP 89LPC925 microprocessor.

LED. The red LED indicates the device is seeing the serial bitstream. The green LED indicates the GPRMC sentences are being seen. The blue LED indicates that the GPS device has achieved lock and is outputting precise time and location.

The software uses a simple state machine to search for the GPRMC sentences in the bitstream. Once this data is located, the software then uses a routine to scan through the comma-delineated fields to extract the three pieces of information. The validity flag will be "V" if the device has not achieved a position solution (lock) and hence is not outputting the time of this solution. This "V" changes to an "A" when the device is outputting the position and time

```
$GPRMC,,V,,,,,,201110,15.6,W,N*1B
$GPRMB,V,,,,,,,,,,,,A,N*13
$GPGGA,,,,,,0,00,,,M,,M,,*66
$GPGSA,A,1,,,,,,,,,,,,,,,*1E
$GPGSV,3,2,12,17,50,280,44,19,14,071,00,20,26,117,00,24,37,051,00*7
E
$GPGLL,,,,,,V,N*64
$GPBOD,,T,,M,,*47
$GPVTG,,T,,M,,N,,K*4E
$PGRME,,M,,M,,M*00
$PGRMZ,,f,1*29
$PGRMM,WGS 84*06
$GPRTE,1,1,c,*37

$GPRMC,151608,A,4255.3513,N,07135.0945,W,0.0,0.0,201110,15.6,W,A*22
$GPRMB,A,,,,,,,,,,,,A,A*0B
$GPGGA,151608,4255.3513,N,07135.0945,W,1,03,3.0,186.3,M,-
32.3,M,,*7B
$GPGSA,A,2,,08,,,17,,,,,28,,,3.1,3.0,1.0*37
$GPGSV,3,3,12,27,14,323,00,28,82,322,44,32,22,095,00,46,36,218,00*7
A
$GPGLL,4255.3513,N,07135.0945,W,151608,A,A*5D
$GPBOD,,T,,M,,*47
$GPVTG,0.0,T,15.6,M,0.0,N,0.0,K*7C
$PGRME,35.2,M,50.0,M,61.2,M*1A
$PGRMZ,611,f,2*1C
$PGRMM,WGS 84*06
$GPRTE,1,1,c,*37
```

Figure 5—These example NMEA sentences illustrate the typical output from a GPS just prior to and just following "lock." The key sentence for the Timeshare system is the GPRMC data, which is the first sentence output in each block from the Garmin system used to capture this data. In the GPRMC data, the lock indicator is the "A" following the second comma. If this "A" is present then the UTC time precedes it (151608 in this case) and the date (201110) follows the ninth comma. Note that date was available prior to lock although the time was not.

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

Unlock TWO Decades of Engineering Excellence
CircuitCellar.com
CC Key fits on any keyring and contains the entire Circuit Cellar issue archive in searchable PDF format.
Keep it current with your Digital 2.0 subscription.
Buy now at www.cc-webshop.com

information. This validity flag is used to trigger the rest of the processing logic that converts the time and outputs it for the slave receivers. These NMEA sentences also include a checksum pair of characters at the end of each line. Since the interface seemed to be solid, this data was not used. If you experience data corruption or want higher confidence in the data, these characters can be used.

MASTER: TIMEKEEPING

Timekeeping seems like an easy problem conceptually, but there are a few speed bumps. Most systems keep time by selecting an epoch start time and then counting fixed increments (seconds, milliseconds, etc.) past this time. Doing math on this time is a simple matter of adding or subtracting increments and then converting the result to an output format (i.e., hours, minutes, seconds, days, months, years). The Timeshare project does not use this approach, but instead performs all calculations using seconds, minutes, hours, days, months, and years. This is somewhat more complex, but arrives at the same end result. This time data is kept in an internal format that is compatible with Maxim's DS1307 RTC. Table 1 is an extract of the DS1307 datasheet describing this data format. The time fields are maintained in binary-coded decimal (BCD) so some logic is devoted to BCD-to-binary conversion and its inverse.

A DS1307 RTC is incorporated in the master unit to enable it to freewheel through loss of lock in the GPS device. The logic could have used this clock for all transmissions to the slave receivers, but for reliability, transmissions are stopped if the outage lasts longer than four hours. The processing recognizes the GPRMC sentences, selects valid data, translates the time and date into binary and BCD, and writes the BCD data to the RTC. Two additional functions, calculating local time and recognizing daylight saving time, need to be added to complete the time processing.

Adjusting for local time using the seconds, minutes, and hours format is a bit challenging. The challenge is not in arriving at the process, but implementing the details. On the East Coast, four or five hours must be subtracted from the UTC value to arrive at a local time. Thinking this process through, if subtracting these hours rewinds the clock past midnight, then the day needs to be stepped back. If the day is stepped back at the first of a month, then the month needs to be adjusted. If the month adjusted is January, then the year needs to be adjusted. All of this needs to be implemented with the variability of 28- to 31-day months, leap years, and the normal challenges of Assembly code. The code that accomplishes this is available on the *Circuit Cellar* FTP site.

```
L23541872711105331U23542372711105331
L24581872711105331U24582372711105331
L25021972711105331U25020012811105332
L26061972711105331U27060012811105332
```

Figure 6—Twelve minutes of data from the Timeshare system are captured in these four transmissions. The "L" indicates local time follows which is then encoded as seconds, minutes, hours, day of week, day, month, year, standard time indicator, and day of year. The "U" then indicates that UTC time follows in the same format. This capture occurred over midnight UTC time, so day changes can be seen in the data.

Having mastered the UTC to local conversion, the device must also check to see if the current time falls within the politically selected intervals for daylight saving time. This process uses the UTC time to check against start and stop dates/time for the changeovers and then select either a four- or five-hour bias for the UTC to local conversion. This is a much simpler process than the UTC to local conversion. The information that is carried in the WWVB signal that flags daylight saving time was also generated. A character of "S," "F," "B," or "D" is added to the transmitted time to indicate standard time, time to change forward, time to change backward, or daylight saving time.

Figure 6 is a sample of four transmissions that are centered about midnight UTC. Local time is transmitted first preceded by an "L." The first time transmitted is 18 hours, 54 minutes, and 23 seconds. The day of week is coded next as a "7." Date follows as 27th day of the 11th month of 2010. The arrangement of this data mirrors the DS1307 format. "S" then indicates standard time and this date is

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Function	Range
00H	CH	10 Seconds			Seconds				Seconds	00–59
01H	0	10 Minutes			Minutes				Minutes	00–59
02H	0	12	10 Hour	10 Hour	Hours				Hours	1–12 +AM/PM 00–23
		24	PM/AM							
03H	0	0	0	0	0	Day			Day	01–07
04H	0	0	10 Date		Date				Date	01–31
05H	0	0	0	10 Month	Month				Month	01–12
06H	10 Year				Year				Year	00–99
07H	OUT	0	0	SQWE	0	0	RS1	RS0	Control	—
08H–3FH									RAM 56 × 8	00H–FFH

Table 1—This format for data in a Maxim DS1307 RTC is extracted from the datasheet. The memory of the DS1307 contains time as seconds, minutes, hours, day of week, day of month, month, and year. The clock also has a control word to select the output of a periodic interrupt and a number of memory cells available to the user. The output of the Timeshare system mimics words 0 to 6 of this format.

Can You Solve This?



JAMECO PUZZLER

The year is 2021, and the Federal Panel of Obsolete Technology (PØT) has banned the manufacture, sale and use of manually adjustable (analog) potentiometers, trimmers and the like. Only digitally-programmable potentiometer technology may be manufactured and used. Consultant Leo Smith is an analog old timer with a thriving business designing work-arounds for circuits that would work much better if "old fashioned" analog components banned by the PØT were still available. How did Leo design a rotary-style, continuous (no steps) volume control for a children's portable radio without breaking the law?

See if you are correct at ***www.Jameco.com/teaserpot***.



Order Your Free Catalog!
www.Jameco.com/catalog

JAMECO® ELECTRONICS

1-800-831-4242 | www.Jameco.com

the 331st day of the year. "U" indicates UTC time which follows in the same format.

MASTER INTERFACING

The master device now has a valid time available and can distribute it to the slaves. Since the transmitter operates as an "Intentional Radiator" under the Federal Communications Commission (FCC) Part 15 regulations, it must meet certain requirements.

Designing a system that operates in this band is easy if the system is only used as a remote control to open a garage door or some similar sporadic task. Sending a periodic signal requires you to meet much more restrictive rules. These rules are designed to make sure all devices using these frequencies limit interference with each other. This means limiting both the power output and the duration of time a device is transmitting.

Many weather-reporting devices incorporate remote wireless capability in these bands and therefore have a low update rate. This rate keeps power consumption low, but it also enables them to meet the restrictions of Part 15 by limiting how often the device is on the air. This project uses the same approach. The datastream consists of the local time and UTC preceded and followed by a 10-ms on period. The initial 10-ms period enables the receiver to synch up on the signal and the ending period enables the serial interface to process the last character sent. To calculate the total On time it is assumed that the transmitter would be on for half the time for each 8-bit word being sent (plus start and stop bits). The On time was then 20 ms plus half the duration for transmitting the local and UTC time. After all these calculations, to be safe, the data is transmitted once every four minutes. This frequency is more than enough to ensure that the remote receivers remain synched and also catch the change to and from daylight saving time.

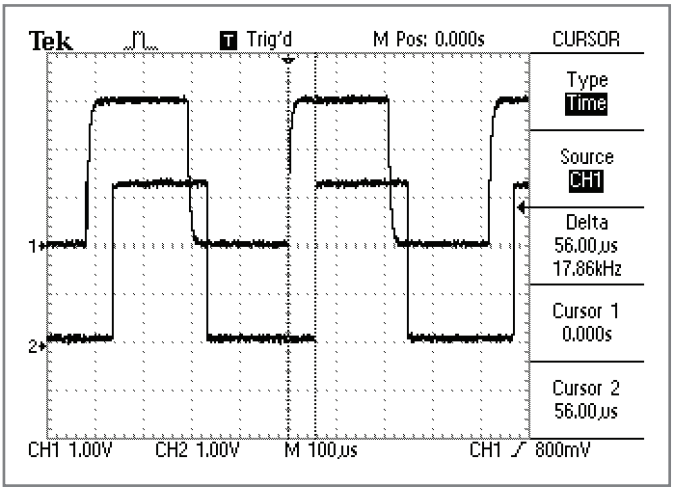


Figure 7—The Linx TXM-418-LC transmitter module has a slight delay when turning on. The datasheet indicates this can vary between 30 and 80 µs. This waveform capture shows the transmitter control as the upper trace and the received signal as the lower trace. The turn-on delay can be seen, as can other delays through the system. Software in the Timeshare system accounts for this turn-on delay to reduce the chance of reception errors.

There is one other feature that needs to be added to the master transmitter to improve signal reception. The Linx On/Off transmitter specifications for the TXM-418-LC transmitters state that they have a maximum data rate of 5,000 bps—the long-range (LR) versions operate at up to 10,000 bps instead of the 5,000 bps for the low-cost (LC) versions.Using 2,400 bps provides significant margin to this maximum data rate. The timing information in the datasheet indicates the transmitter turn-on time is a maximum of 80 µs and a turn-off time is 100 µs.

Figure 7 validates this behavior by looking at a square wave as it is transmitted and received by the Linx equipment. The turn-on time delay will shorten a "one" if it follows a zero. To reduce the chance of error in the communication, the time periods for the On and Off periods are calculated in software rather than using the hardware serial interface. With this control, if a "zero" precedes a "one" pulse then the "one" can be extended by the 80 µs to compensate for the turn on delay. A "one" pulse preceded by another "one" does not need to be modified.

Adjusting these timings in software produced an accurate 2,400-bps signal on the receiving end. It may not have been absolutely necessary, but at the range limits of the receiver/transmitter pair, and with additional electrical noise in the environment, it never hurts to give the system some additional margin. In this case, the software did this for free.

SLAVE INTERFACING

The receiver is the simple part of this project. A small microprocessor accepts the ASCII output stream from the Linx receiver and writes it to the target's DS1307 RTC across an I²C interface. The only significant task the microprocessor has to perform is error checking the input stream. OOK transmission systems are subject to significant interference and can generate reasonable looking data from noise.

The microprocessor uses the built-in error checking of the universal asynchronous receiver/transmitter (UART) serial interface to remove some of the errors and then performs additional checking on the reasonableness of the data before passing it to the I²C interface. These checks pretty much ensure that only valid time signals are used to update the target's clock. These checks also mean that some signals may be discarded if they are corrupted. Since the time is transmitted every four minutes, there will be other opportunities to extract a valid time word.

Testing of the completed system has shown excellent reception of valid data at multiple locations in a structure. Even with noise present, a valid time word is extracted when the transmitter and receiver are separated by two floors and multiple walls. If more confidence is required, checksums could be added to the data or portions of the local time fields could be compared to the UTC time fields. If a higher repetition rate is desired, the UTC fields could be eliminated to reduce the On time for the signal and the rate could be

increased to one transmission every two minutes.

The red/green/blue LED in the receiver is used to indicate the status of this input stream. Red indicates that data has not been seen while blue is used to indicate that a valid time transmission has been received. The LED is blinked if the system detects noise on the data stream. The Linx RXM-418-LR receivers are 10 to 20 dBm more sensitive than the low-cost RXM-418-LC receivers.

Although these low-cost receivers are less sensitive, they did the job well and had very little noise. The long-range ones require more caution.

Once the data is available to the I²C interface, the problem of multiple bus masters is encountered. In many projects, the microprocessor is the only bus master for the I²C interface to a slave RTC chip and to multiple other I²C slave devices. This is a well-ordered world, one master controlling multiple slaves. A multiple-master system needs to employ additional logic to see if two devices are trying to use the bus at the same time and seize and release the bus appropriately.

There are three basic processes going on to accomplish this. The first is making sure the clocks from the two masters are in sync. This is accomplished by the low periods of the clocks being stretched to match the slowest device on the bus. The high periods are shortened to match the fastest device. This logic works this way because the clock line is actively pulled low and can float high with the help of pull-up resistors. The slowest device to release the low on the clock line is the one that actually accomplishes the change from a zero to a one. The first device to pull a high clock to a low will be the one to determine the high period.

Once the clocks are in sync, the second basic process is tracking the state of the bus. The bus can be free or busy. The busy state is everything between a start and a stop command. No device can start activity on the bus if it is busy with another device. This logic is somewhat complex if attempted in a software-only solution since all transitions of the data line need to be tracked to see if they occurred during a high on the clock. These types of transitions signal the start and stop conditions.

The third process is arbitration, where two or more devices have started a transaction on the bus at the same time. These transactions continue until one device transmits a zero when another transmits a one. The device transmitting a zero continues to keep the bus and the other device must back off until the bus becomes free again, then restart its transaction. It is possible to imagine a sequence where two devices complete identical transactions not knowing that another device is even on the bus. In most cases though, the datastreams will diverge at some point and the arbitration logic must step in to remove one device from the bus.

Most of this complex logic to handle these conditions is provided by the





LPC925 microprocessor hardware. The software only needs to recognize specific indicators provided by the hardware and react appropriately. In the software for this project, this arbitration logic is captured in the library element (I2C_MM.inc) which interprets the interrupt conditions from the hardware. Once a device starts a transaction, it will complete normally unless the slave device does not respond at all.

The I²C interface is not the only option for the receiver. The output of the Linx device is a standard 2,400-bps ASCII stream, so the project's microprocessor could read this stream, perform error checking, and update its internal clocks. I²C was used for the Timeshare project since the target projects had already incorporated I²C and not all had an available serial interface.

The I²C interface implemented in the target projects has been standardized to use an RJ-45 jack/plug for both the signal and the power connections to remote devices. Each project incorporates a series of RJ connectors. This enables multiple I²C slave devices to be plugged into the project with a standard interface. The RJ interface also provides an inexpensive implementation option. A number of surface-mount boxes are sold for both telephone and data cable connections. These phone boxes are available at hardware stores and are easily modified to hold a circuit board. The cases have a snap-off lid, an RJ connector with four wires, and room inside for a few integrated circuits. Some of these boxes come with twin connectors so the I²C interface can be daisy chained through multiple devices.

All of these configurations need attention to proper drive and termination of the I²C signals. I²C documentation obtained from Texas Instruments provides great guidance on this. An I²C driver at the beginning and end of the string of devices works quite well with microprocessors for the intermediate devices. Other I²C devices will require an interface chip since they most likely will not be able to sink the current on the extended bus. The 89LPC925 can sink about 20 mA into port pins and can also float pins to meet the I²C specifications. The Timeshare project implements the I²C directly using these port pins from the microprocessor.

SYSTEM CHECKOUT

Putting the pieces together for this project is fairly easy since most of the interfaces use a standard serial bitstream. The output from the GPS receiver can be read using HyperTerminal or similar software on a PC. This verifies the bitstream is as expected and that the data rate is correct. The output from the microprocessor to the transmitter is also a serial bitstream, which can be converted to a standard RS-232 signal using a chip such as the MAX202. On the receiver side, the output from the Linx receiver can be converted to RS-232 by the same converter chip and then displayed on the PC. If you elect to use the software supplied, keep in mind that the usual cautions apply: one bug is present in all systems and when that one is removed there is another. Most of the bugs in this system will probably be found lurking in the conversion of the time bases, although checking indicates the code is working correctly.

THE FUTURE

There are many options available for future growth of the Timeshare system. With the addition of a battery back-up and some additional code, the microprocessor in the receiver could mimic the behavior of the DS1307. Once these changes are made, the clock chips can be removed from the projects without modifying the hardware or software in them. For now, though, all the projects within receiving range of the Timeshare transmitter can be easily and reliably synchronized to one time source.

Dave Penrose (david.penrose@comcast.net) is retired from the aerospace industry. He began his career programming test software for space vehicles and has maintained a lifelong passion for developing microprocessor projects. Dave holds an Expert Class Amateur Radio License (K1DHP) and earned an MS in Computer Science from the University of Santa Clara and a BA in Mathematics from the University of California at Berkeley. He holds a firm belief that the hours between midnight and 5:00 AM should be used for sleeping rather than debugging with a signal from WWVB, hence this article's focus on GPS time. Much credit for putting up with strange devices appearing in the house and for editing rough thoughts into finished articles goes to Dave's wife, Melinda.

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2012/258.

RESOURCES

Federal Communications Commission (FCC), "Electronic Code of Federal Regulations (e-CFR), Title 47: Telecommunication, Part 15—Radio Frequency Devices," 2011, http://ecfr.gpoaccess.gov.

Maxim Integrated Products, "DS1307 64 × 8, Serial, I²C Real-Time Clock," 2011, www.maxim-ic.com/datasheet/index.mvp/id/2688.

Texas Instruments, Inc., "P82B715 I²C Bus Extender," 2008, http://focus.ti.com/lit/ds/symlink/p82b715.pdf.

SOURCES

TXM-418-LC Transmitter module, RXM-418-LR receiver, and RXM-418-LC receiver
Linx Technologies | www.linxtechnologies.com

MAX202 Transceiver and DS1307 real-time clock
Maxim Integrated Products | www.maxim-ic.com

89LPC925 Microprocessor
NXP Semiconductors | www.nxp.com

BitScope

Digital + Analog

Mixed Signal Oscilloscopes

100 MHz Digital Oscilloscope

✓ Dual Channel Digital Storage Oscilloscope with up to 12 bit analog sample resolution and high speed real-time waveform display.

40 MSPS x 8 Channel Logic Analyzer

✓ Captures eight logic/timing signals together with sophisticated cross-triggers for precise multi-channel mixed signal measurements.

Serial Logic and Protocol Analyzer

✓ Capture and analyze SPI, CAN, I2C, UART & logic timing concurrently with analog. Solve complex system control problems with ease.

Real-Time Spectrum Analyzer

✓ Display analog waveforms and their spectra simultaneously in real-time. Baseband or RF signals with variable bandwidth control.

Waveform and Logic Generators

✓ Generate an arbitrary waveform and capture analog & digital signals concurently or create programmable logic and/or protocol patterns.

Multi-Channel Chart Recorder

✓ Record to disk anything BitScope can capture. Allows off-line replay and waveform analysis. Export captured waveforms and logic signals.

Everything in one tiny package !



BitScope Pocket Analyzer is the world's first Mixed Signal Scope to include a powerful Logic Protocol Analyzer, Waveform & Pattern Generator, Spectrum Analyzer and Chart Recorder in one ultra-portable **USB powered** package.

Pocket Analyzer is fast, displaying up to 100 frames per second with up to 12kS per frame. Alternatively stream data direct to disk for replay and analysis.

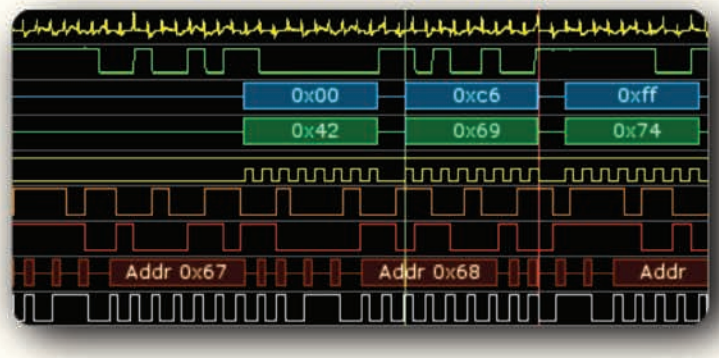
Compatible with major operating systems including Windows, Linux & Mac OS X, Pocket Analyzer is your ideal test and measurement companion.



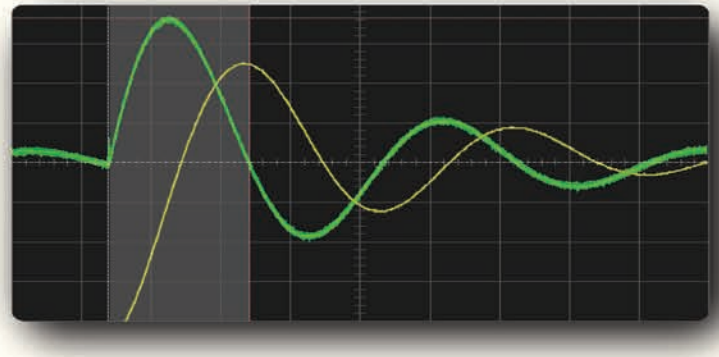
Pocket Analyzer



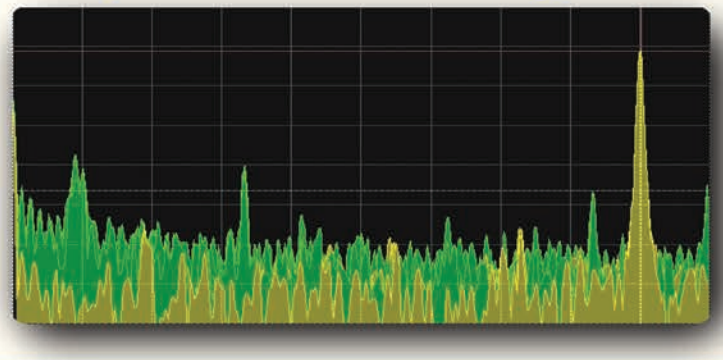
Protocol Analyzer



Digital Oscilloscope



Spectrum Analyzer



bitscope.com/cc

by Andrew Levido (Australia)

mbos

A Real-Time Operating System for mbed

This project enhances the NXP Semiconductors mbed environment with a real-time operating system (RTOS). Designed specifically for mbed, "mbos" is a true preemptive multitasking RTOS that's scalable to support virtually any project developed on the NXP Semiconductors LPC1768 platform.

If you're like me, the first microcontroller you used was one of the venerable 8-bit architectures with just a handful of basic on-board peripherals. The datasheet was a few dozen pages long, and it did not take too long to master the device. How things have changed! We now have a plethora of low-cost, 16- and 32-bit chips, each with an alphabet soup of on-board peripherals (e.g., Ethernet, CAN, USB, DMA, etc.). The downside is that the learning curve for each new device has become very steep. New users need to wade through thousands of pages of documentation and install and learn a complex toolchain before they can begin to exploit the incredible power and flexibility of these devices.

While looking at some of the newer 32-bit microcontrollers, I came across the mbed website and was impressed with their unique approach to this problem. On the hardware side, the mbed module includes an NXP Semiconductors LPC1786 microcontroller, which is based on the ARM Cortex M3 architecture and includes just about every peripheral you could name. The real innovation is the user-friendly online development environment and a library of drivers that simplify the task of using the peripherals, hiding most of the complexity from the user. Members of the mbed online community have published additional libraries of drivers for a bewildering variety of external devices such as displays, servos, and EEPROMs. I ordered my mbed module on the spot.

Once I started to play around, I saw that while mbed is designed to hide unnecessary complexity, it does not dumb down the microcontroller in any way, so it is suitable for developing very large and complex applications. The libraries can significantly shorten your development time, but they can't eliminate the usual embedded systems challenge of ensuring that multiple asynchronous real-time activities don't trip over each other. This is the point where I normally look for a real-time operating system (RTOS).

RTOS

An RTOS enables a user to divide an application into discrete tasks, each of which runs independently of the others, as if it had the processor to itself. A scheduling algorithm determines when each task can run. Since the tasks are isolated from each other, the RTOS must also provide a signaling mechanism to enable the tasks to communicate with each other, and to access shared microcontroller resources (such as peripherals) without interference from other tasks. Since the interaction between tasks and between tasks and the outside world is tightly regulated, it is relatively easy to write code that responds in a predictable way to the chaotic unpredictable real world.

There are many RTOSes available for the M3 processor, ranging from some excellent free ones to some very expensive proprietary systems. They all share one characteristic: complexity. Porting one to the mbed is a challenging task, especially since the debugging facilities within the mbed environment are rudimentary. Anyway, this approach is not compatible with the mbed philosophy of keeping the complex details hidden from view.

Public Member Functions

	mbos (uint ntasks, uint ntimers=0, uint nresources=0) Create an mbos object.
void	**Start** (uint idlestacksize=32) Start mbos.
void	**CreateTask** (uint taskid, uint priority, uint stacksz, void(*fun)(void)) Create an mbos task.
uint	**GetTask** (void) Get the ID of the current task.
void	**SetPriority** (uint priority) Set the priority of the current task.
uint	**GetPriority** (void) Get the priority of the current task.
void	**WaitEvent** (uint event) Wait for an event or events.
void	**SetEvent** (uint event, uint task) Post an event or events to a task.
uint	**GetEvent** (void) Returns the event flag(s) which last caused the task to unblock.
void	**CreateTimer** (uint timerid, uint taskid, uint event) Create a mbos timer.
void	**SetTimer** (uint timerid, uint time, uint reload=0) Starts an mbos timer.
void	**RedirectTimer** (uint timerid, uint taskid, uint event) Redirects an mbos timer.
void	**ClearTimer** (uint timerid) Stops and clears an mbos timer.
void	**CreateResource** (uint resourceid, uint priority) Creates an mbos resource.
uint	**LockResource** (uint resourceid) Locks an mbos resource and temporarily allocates the resource's priority to the calling task.
uint	**TestResource** (uint resourceid) Tests whether a resource is locked or free, without changing its state.
uint	**FreeResource** (uint resource) Frees a resource Frees an mbos resource and restores the calling task's original priority.

Figure 1—The complete API for mbos consists of just 17 functions. A full description of the API is included in the API documentation in the design files.

Hence my decision to create a new RTOS called "mbos," which is specifically designed for the mbed environment and conforms to the mbed design paradigm.

SETTING OBJECTIVES

As usual, at the start of any project, I set myself some goals. I wanted mbos to be a true preemptive multitasking operating system (OS). Cooperative scheduling (where tasks must voluntarily give up control of the processor) works just fine, but for an application to respond quickly to real-world events, preemption is a must.

I needed mbos to be scalable so that it would cope with the most complex application that could conceivably be squeezed into the LPC1768 microcontroller. On the other hand, the OS had to be light enough to make sense for modest applications with just two or three tasks, which is often all that is needed.

I planned mbos to be frugal with the microcontroller's resources—especially the RAM, which, at 32 KB for general purposes, is generous but by no means unlimited. To be frank, this goal was not too much of a challenge. I cut my microcontroller teeth in the days when RAM was measured in bits!

I also wanted the mbos kernel to be reasonably fast. I wanted to leave the maximum possible number of cycles for the application, so scheduling and context switching would have to use as few cycles as possible. This meant some of the kernel would have to be written in Assembler, and a wise choice of scheduling algorithm would be important.

And, finally, I wanted to ensure that the API would be as simple as possible, but not simplified to the point that functionality and flexibility were compromised. This turned out to be the hardest balance of all to achieve (see Figure 1).

I think the end result meets all of these criteria. The process of developing and testing mbos has been an enjoyable one, involving a myriad of design decisions. In this article, I describe the key decisions—those that give mbos its unique characteristics—in detail.

TASKS & TASK SWITCHING

To keep things simple, I decided tasks should be static and be limited to two possible states. Tasks are created before the kernel is started and are either "ready" for their turn on the processor when their priority dictates, or they are "waiting" for some external event. As well as a task function, each task requires a dedicated stack in RAM and a simple data structure called the task control block (TCB) to store the task's ID, priority, state, and the event(s) it is waiting for or has received. We also need to store the stack pointer when the task is not running and a pointer to the top of the stack so we can test for stack overflow. Listing 1 shows the TCB structure. Notice that the priority and state are combined into one variable called `priostate`. More on this later.

The most important job of the RTOS kernel is to manage the swapping of tasks into and out of the processor in such a way that a swapped-in task resumes execution exactly where it left off. The Cortex M3 processor designers anticipated the use of an OS and thoughtfully provided three processes or exceptions that can be used to implement an RTOS: `SysTick`, `SVCall`, and `PendSV`. Exceptions are the generic term for any event that asynchronously interrupts the processing of instructions and redirects execution to a specific piece of handler code. Interrupts are one type of exception.

I used the `PendSV` exception to trigger the context switch. When it occurs, the Cortex M3 processor automatically stacks processor registers R0–R4, R12, the Link register, the return address, and the Status register. This means all we have to do to save the entire context is

Listing 1—The task control block data structure. Note that the task's priority and state are combined in the `priostate` variable. The `stacklimit` variable is used to check for stack overflow.

```
typedef struct _tcb{       // Task control block
      uint id;
      uint *stack;
      uint priostate;
      uint eventlist;
      uint eventmask;
      uint *stacklimit;
}_tcb_t;
```

Listing 2—Saving and restoring the context only takes a couple of lines of Assembler code. Most of this code is concerned with saving and retrieving the stack pointers and checking for stack overflow.

```
PendSV_Handler                              ; entered here for context switch
     cpsid          i                       ; disable interrupts

     ; save context of running task (those not saved by exception entry)
     mrs            r0, msp
     stmdb          r0!, {r4-r11}

     ; store the sp in the tcb
     ldr            r1, =_tasklist
     ldr            r1, [r1]
     ldr            r1, [r1]
     str            r0, [r1, #STACK_OFFSET]

     ; check for stack overflow against task limit
     ldr            r2, [r1, #LIMIT_OFFSET]
     cmp            r0, r2
     bls            stackerror

     ; check for stack overflow against hard limit
     ldr            r2, =_hardstacklimit
     ldr            r2, [r2]
     cmp            r0, r2
     bls            stackerror

SVC_Handler                                 ; entered here for 1st task start
     cpsid                                  ; Disable interrupts

     ;
     ; Sort code here
     ;

     ; restore the context (those not restored by the exception exit)
     ldr            r0, =_tasklist          ; get the sp from the TCB
     ldr            r0, [r0]
     ldr            r0, [r0]
     ldr            r0, [r0, #STACK_OFFSET]
     ldmia          r0!,{r4-r11}            ; pop the stack
     msr            msp, r0                 ; and put the sp away
     cpsie          i                       ; re-enable interrupts

     ; Force a return from exception and the restoration of the rest of the regs
     bx             lr
```

stack the remaining few registers and store the stack pointer away in the TCB. All of this is done in the PendSV exception handler which is written in Assembler as shown in Listing 2. As you can see, restoring the context is the reverse process.

After saving the context and storing the stack pointer (SP), we test for stack overflow. The SP is first tested against the limit for that stack that is stored in the TCB. There is still a remote chance that the stack has overflowed into the TCB invalidating the first test, so we also test the SP against a second hard-coded limit that detects this unlikely but catastrophic scenario.

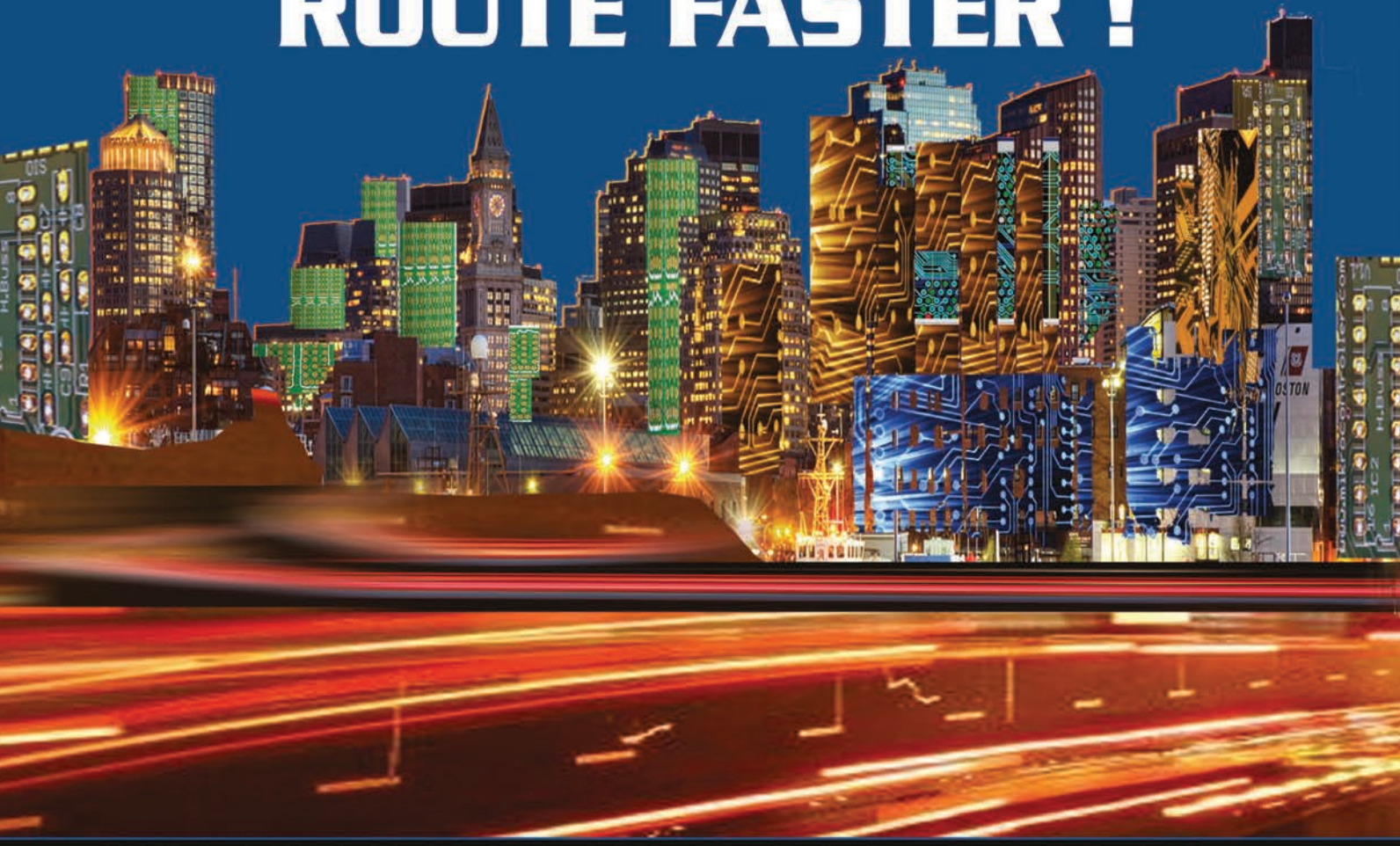
I used the SVCall exception to start the scheduler for the first time. It shares the same handler as PendSV, but enters it after the context save since there is initially no running task to swap out. You can see this entry point about halfway down Listing 2. When each task is created, an empty context is set up in the task's stack, to be "unpacked" the first time the task runs. The initial context includes the task function address so that when the task is first run, execution commences at the start of the task function.

SCHEDULER

The next task was to select a scheduling algorithm that ensures the highest-priority ready task is run. The usual approach is to manage several linked lists, including one for ready tasks. As tasks unblock, they are inserted into the list such that the list remains sorted by priority. This is an efficient mechanism for large lists, but for short lists (say 10 tasks), the overhead of maintaining linked lists becomes significant.

I chose to simply maintain one list of tasks as an array of pointers to the TCBs, and to sort this list in order of the priority of the tasks. I force ready tasks to the top of the list by adding an offset to their priority. Any task that is ready will therefore always have a higher priority than any waiting task. When a task blocks, I remove the offset and the task will fall toward the end of the list the next time it is sorted. This effectively combines the state

ROUTE FASTER !



WITH PROTEUS PCB DESIGN

Our completely new manual router makes placing tracks quick and intuitive. During track placement the route will follow the mouse wherever possible and will intelligently move around obstacles while obeying the design rules.

All versions of Proteus also include an integrated world class shape based auto-router as standard.

PROTEUS DESIGN SUITE

Features:

- Hardware Accelerated Performance.
- Unique Thru-View™ Board Transparency.
- Over 35k Schematic & PCB library parts.
- Integrated Shape Based Auto-router.
- Flexible Design Rule Management.
- Polygonal and Split Power Plane Support.
- Board Autoplacement & Gateswap Optimiser.
- Direct CADCAM, ODB++, IDF & PDF Output.
- Integrated 3D Viewer with 3DS and DXF export.
- Mixed Mode SPICE Simulation Engine.
- Co-Simulation of PIC, AVR, 8051 and ARM7.
- Direct Technical Support at no additional cost.

Visit our website and use Promotional Code CLR2011JGB for an extra 10% Discount. Prices fromt just $249!

labcenter Electronics www.labcenter.com

Labcenter Electronics Ltd. 411 Queen St. Suite 201, Newmarket, Ontario, Canada
Toll Free 866.499.8184, www.labcenter. com or Email: info@labcenter-electronics.com

and priority in one variable and eliminates the need to manage linked lists.

Figure 2 shows how this works. In this example, there are four user tasks that have ascending priority. Tasks 3 and 4 are blocking, waiting for some event. Tasks 1 and 2 are ready, so they have an offset of 100 added to their priority in the `priostate` variable. The kernel sorts the array of pointers (`_tasklist`) in descending order of `priostate`, as shown. Now Task 2, the highest-priority ready task, has been sorted to the top of the list and will be swapped into the processor.

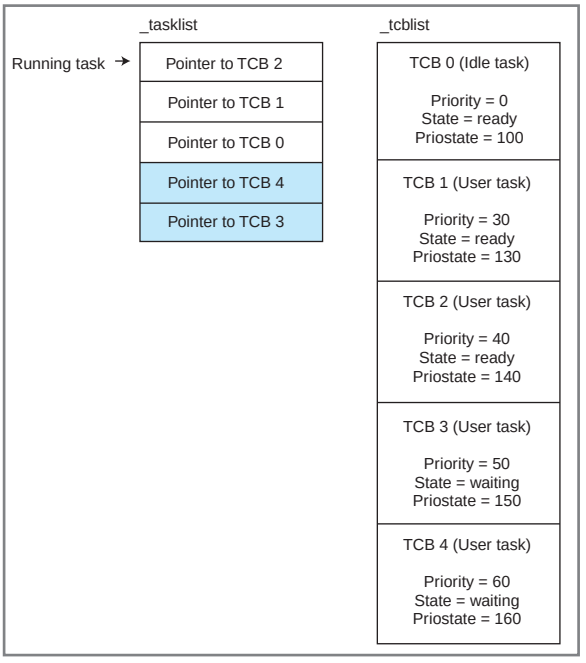


Figure 2—In this example, high-priority tasks 3 and 4 are waiting for an event. Tasks 1 and 2 are ready and therefore have an offset added to their priostate. The sorting algorithm ensures that the highest-priority ready task will always run. The system-generated idle task is always ready.

I selected an insertion sort algorithm because it is one of the more efficient sort-in-place algorithms for somewhat short lists and because it can be implemented so that elements of equal value within the list change position on subsequent sorts. This characteristic is essential to ensure that when tasks share the equally highest priority, each is allocated equal processor time. This is known as round-robin scheduling. Listing 3 shows this algorithm implemented in pseudocode. For mbos, I wrote it in Assembler using processor registers for storing intermediate values to optimize the speed.

The scheduler is invoked whenever a task changes state or when a given time has elapsed. The time period should be short enough so that round-robin tasks really appear to be running in parallel to most real-time events, but long enough so that the kernel does not steal too much processing time from the user tasks with too-frequent swaps. I settled on 1-ms ticks, which puts the kernel overhead at less than 1%, yet enables good real-time performance. The SysTick timer and associated exception are used to generate this, leaving all of the microcontroller's timers free for the user application.

DEBUGGING

With these decisions made, I was ready to code and test the kernel. Here I ran into my first hurdle and exposed the one significant weakness of the mbed development environment—an almost complete lack of debugging facilities. I wanted to single-step through the code and to view and change variables at will. The online mbed compiler just does not support this level of development. I do hope that better debugging facilities are planned for a future version.

Fortunately, there is a workaround. You can download a limited version of the Keil µVision4 ARM Cortex M3 toolchain for free, and, by following the instructions in Keil Application Note 207 (available from their website), you can compile, run, simulate, and debug mbed applications in an "industrial-strength" integrated development environment (IDE). Once I got this up and running, it was a fairly easy task to validate the basics of the mbos kernel.

TASK COMMUNICATION

With task switching and scheduling reliably working, it was time to decide how the tasks should communicate with each other and the outside world. The options are almost endless, with some RTOS implementations providing queues, binary semaphores, counting semaphores, recursive semaphores, and mailboxes, among others. All of these have their uses, but it seemed to me that it would defeat the purpose of mbos if I was to overcomplicate matters. I wrestled with this dilemma for a while, before taking a step back and asking myself what I was really trying to achieve.

When it really comes down to it, the purpose of all these mechanisms is to provide a controlled method for the tasks—which we have gone to a great deal of trouble to isolate from each other—to respond to asynchronous events in their environment and to sensibly share system resources. I thought that if I provided a simple but powerful mechanism for achieving these two tasks, they would suffice on their own yet enable the user to build

Listing 3—The Sort algorithm scans the list searching for the largest item which it then inserts into the first position. It repeats this process for the second position, and so on, until the list is fully sorted.

```
// n items in the list to sort, indexed 0 to n - 1

for i = 0 to n - 2

        k = i

        for j = i + 1 to n - 1

                if item[j] >= item[k]

                        k = j
                end if
        next j

        swap item[i] and item[k]

next i
```

Listing 4—The timer control block data structure. If the reload value is zero, the timer executes once. If the value is nonzero, the timer posts events continuously at an interval determined by the reload value.

```
typedef struct _timer{          // Timer control block
     uint timer;
     uint reload;
     uint event;
     uint task;
}_timer_t;
```

more complex constructs if desired. The mechanisms I chose for managing events and resources were creatively named "events" and "resources," respectively.

Events are just that—asynchronous events that can be posted to a waiting task to signal something has happened (e.g., a timer expiring, some data ready in a peripheral register, an interrupt occurring, etc.). I allocated each task a one-word (32-bit) event mask with each bit representing a separate event. When a task blocks to wait for an event, the appropriate bit in the event mask is set and the task enters the waiting state. A task can wait for more than one event if more than one bit in the mask is set.

When an event is posted to a waiting task, it is compared with the mask. If there is a match, the task is changed to the ready state. The event is captured in the event list so that the task can later determine which event occurred. I provided one function to force a task to block to wait for one or more events, and one to post events to a task. These two functions are all that is necessary to synchronize tasks in even the most complex applications.

Events take care of the communication between tasks and the synchronization of tasks with external actions, but they do not help when it comes to the problem of multiple tasks accessing shared resources such as on- or off-chip peripherals or global memory. If a task is swapped out while it is in the middle of an operation with such a resource, and another task commences to operate on the same resource, the results will be unpredictable and probably catastrophic.

We need a mechanism that enables tasks to temporarily "lock" resources so that competing tasks cannot access them until the operation is complete. Many RTOSes provide "mutual-exclusion semaphores," or "mutexes," for this purpose. This is basically a flag that indicates that a resource is in use. Each task must check the state of the flag to ensure the resource is free before using it. The task sets the flag when taking over the resource and clears the flag when it relinquishes it.

While easy to implement, mutexes suffer from two significant problems. The first is deadlock, which occurs when two or more tasks are each





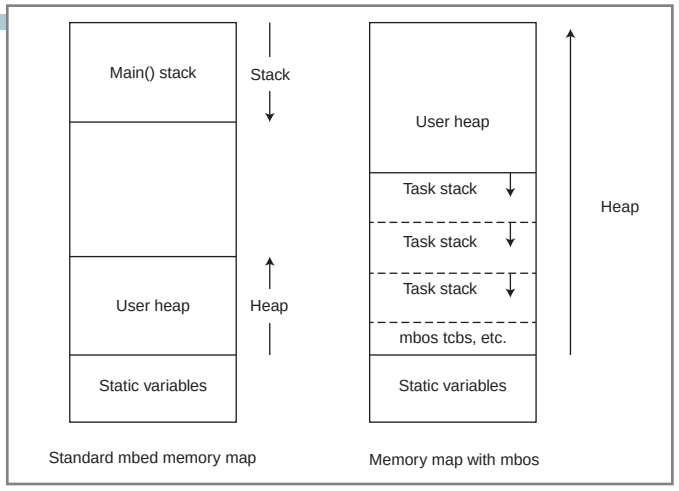


Figure 3—mbos creates the control blocks and task stacks on the heap. Once mbos is started, the main stack is not required, so all the remaining memory is available to the heap. The kernel checks for task stack collision, which can be fatal.

waiting for each other to release a resource, or more than two tasks are waiting for resources in a circular chain. Under these conditions no task can run, and the system grinds to a halt.

A more subtle problem is priority inversion. Consider the situation where a low-priority task has acquired a resource and a high-priority task is waiting for it—all perfectly normal, and in due course, the low-priority task will finish with the resource and the high-priority task will run. But, if a medium-priority task becomes ready while the high-priority task is waiting, it will displace the low-priority task, and the resource will remain locked. Only when the medium-priority task blocks will the low-priority task resume and completely release the resource to the high-priority task. The medium-priority task has effectively trumped the high-priority task; not what the designer had planned.

The solution is a mechanism known as the ceiling protocol. Under this protocol, the resource itself is assigned a priority that is higher than the priority of any task that will access it. When a task locks a resource, it is temporarily allocated the priority of the resource, ensuring that it cannot be swapped out by any of the competing tasks. Once the first task has finished with the resource, its original priority is restored. The ceiling protocol is immune from deadlocks and priority inversion. mbos supports up to 100 resources and the API provides functions to create, lock, free, and test resources.

TIMERS

Most applications require time delays or periodic actions. With an RTOS, we want to make sure that when a task calls a delay it is swapped out for the duration so that other tasks have an opportunity to run and the processor cycles are not wasted. Since we already use the System Tick to provide the basic timeslice for the scheduler, it is an obvious step to use it to create a timer mechanism. Each timer consists of a small data structure as shown in Listing 4. When a timer is created, a timer data structure is populated with the ID of the task to which the timer will post an event on time out, and the event to post. The timer and reload members are zeroed at this stage.

When the timer is started, the data structure is loaded with an initial (nonzero) timeout value and an optional reload value for subsequent cycles. Each system tick, the nonzero timer members of all timers are decremented. If any reach zero, the appropriate event is posted and the reload value is transferred to the timer variable.

If the reload value is zero, the timer will post the event once after the initial time period has expired and then the timer will halt. If the reload value is nonzero, the timer will continue to post events indefinitely (until cleared or reset) each reload period. Thus, one timer mechanism can do duty as a one-shot or continuous timer. The API provides functions to create, redirect, set, or clear timers.

MEMORY MANAGEMENT

The mbed module implements a single memory-space model, where the heap and the stack build from opposite ends of memory. Statically, allocated variables are located in the lowest part of the RAM, and the space above this is allocated to the heap, which grows toward the top of the RAM. At the same time, the stack pointer is initialized to the top of the RAM and the stack builds downward. It is up to the user to make sure they don't collide with each other, or to deal with the consequences if they do collide. Refer to the left side of Figure 3.

mbos allocates a few static variables in the lower part of the RAM, but all the memory required for tasks and their stacks, timers, and resources is allocated dynamically from the heap when they are created. Once mbos starts, the original system stack is no longer required, since one or another of the task stacks will always be used instead. The entire heap is therefore available to the user's application as shown on the right side of Figure 3. There is no longer any risk of the heap colliding with the stack, but instead a risk that one stack will grow into another. This is why the kernel performs the rigorous stack overflow checking previously described.

IMPLEMENTATION

I wrote the mbos RTOS in a combination of C and Assembler. To make it a publishable mbed library, I enclosed it in a C++ "wrapper." I built the software from the start with solid error detection and reporting. Each function tests for illegal parameters and takes the appropriate action. In most cases, this means flagging a run-time error, but occasionally it is more appropriate for the function to return and do nothing. For example, if you post an event to a valid task that is not currently waiting for that event, nothing happens. If you post an event to a nonexistent task, an error is flagged.

I made use of the error function from the mbed standard library to deal with run-time errors. For each error, an



▶ CUWIN6300

▶ CUWIN5500

▶ CUWIN5200

Windows CE 6.0 Touch Controller

CUWIN

The CUWIN is a series of Windows CE touch controllers that are more cost-effective than a PC, but with more features than an HMI touch screen.
Create sophisticated applications with C++ or any .Net language.

533MHz ARM CPU
128MB SDRAM & Flash
SD Card Support
Ethernet, RS-232/485
USB, Audio Out
Windows Embedded CE 6.0



C-Programmable Modular Industrial Controller

MOACON

The MOACON is a modular, C programmable, ARM-based automation controller designed for industrial environments.

Choose from a diverse, feature-rich selection of modules including:
Digital I/O
Analog I/O
RS-232/485
Motor Control
Ethernet
High-speed Counter & PWM

BASIC with LADDER LOGIC CONTROLLER



New

Integrated CUBLOC Controller and I/O Board

CB210

The CB210 is an inexpensive, integrated CUBLOC controller and I/O board programmable in both BASIC and Ladder Logic.

I/O Ports x20	10-bit A/D x6
PWM x3	RS-232 x1
80KB Program Memory	3K Data Memory

The CUPC is a series of industrial touch panels with all the features of a modern PC for the most feature-rich user experience.

ATOM N270 1.6GHz CPU
2GB RAM
320GB HDD
SD Card Support
Color Display (1024 x 768)
RS-232, Ethernet
USB, Audio Out
Windows XPe/XP Pro

CUPC

Industrial Touch Panel PC with ATOM Processor



▶ CUPC-P200

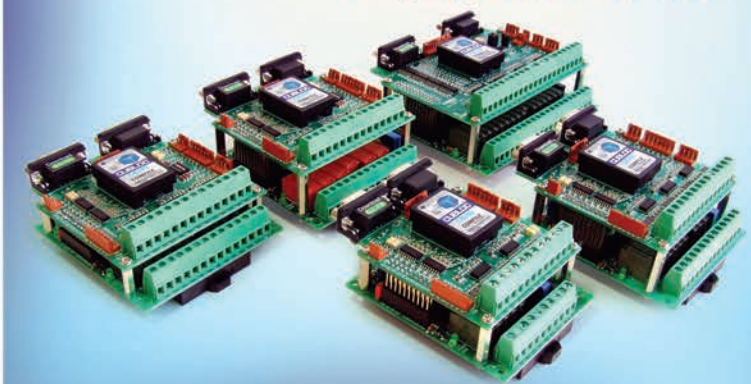
▶ CUPC-P220

The CUSB is a series of compact, CUBLOC-integrated, industrial I/O boards programmable in both BASIC and Ladder Logic.

CUBLOC CB280 Core Module
80KB Program Memory
3KB Data Memory
DC 24V Inputs x9~16
A/D Inputs x2~6
Relay Outputs x6~16
PWM x2~6
High-speed Counters x0~2
RS-232/485 x1~2

CUSB

Integrated Industrial Controllers



COMFILE TECHNOLOGY
1175 Chess Dr., Suite F, FOSTER CITY, CA 94404
call : 1-888-9CUBLOC (toll free)
1-888-928-2562
email : sales@comfiletech.com
www.comfiletech.com

explicit error message is posted to the C++ stderr stream (by default the USB serial port), and the code exits gracefully and displays the familiar mbed "blue lights of death." A summary of the API functions is shown in Figure 1. The full API is documented in the API reference included in the design files.

A FRAMEWORK TO BUILD ON

Comparing the result with my original objectives, I have achieved everything I set out to do. In a typical application the kernel takes 5.5 µs to save a task, determine which task to run next, and restore that task's context. This amounts to only 0.6% of the available processor cycles, leaving 99.4% for the user's application.

Only 36 bytes of RAM are required for the kernel, plus 28 bytes per task, 16 bytes per timer and 12 bytes per resource. A huge application with 20 tasks—each with a 128-word stack, 20 timers, and 20 resources—will occupy

Listing 5—This simple example demonstrates the ease with which you can write an RTOS application. Task 1 toggles an LED and then sends a message to Task 2 each time a timer posts an event. Task 2 briefly flashes a second LED when it receives the message.

```
// mbos Blinky demonstration.
// Task 1 toggles LED1 every second, under control of a timer. It then posts an event to
// task 2 which flashed LED2 briefly.
#include "mbed.h"
#include "mbos.h"

#define TASK1_ID              1       // ID for task 1 (idle task is 0)
#define TASK1_PRIO            50      // priority for task 1
#define TASK1_STACK_SZ        32      // stack size for task 1 in words
#define TASK2_ID              2       // ID for task 2
#define TASK2_PRIO            60      // priority for task 2
#define TASK2_STACK_SZ        32      // stack size for task 2 in words
#define TIMER0_ID             0       // ID for timer 0
#define TIMER0_PERIOD         1000    // Time period in milliseconds
#define TIMER0_EVENT          1       // Event flag (1 << 0)
#define T1_TO_T2_EVENT        2       // Event flag (1 << 1)

void task1(void);                     // Task function prototypes
void task2(void);

DigitalOut led1(LED1);
DigitalOut led2(LED2);
mbos os(2, 1);                        // Instantiate mbos with 2 tasks & 1 timer

int main(void)
{
    // Configure tasks and timers
    os.CreateTask(TASK1_ID, TASK1_PRIO, TASK1_STACK_SZ, task1);
    os.CreateTask(TASK2_ID, TASK2_PRIO, TASK2_STACK_SZ, task2);
    os.CreateTimer(TIMER0_ID, TIMER0_EVENT, TASK1_ID);
    // Start mbos
    os.Start();
    // Never  return!
}

void task1(void)
{
    os.SetTimer(TIMER0_ID, TIMER0_PERIOD, TIMER0_PERIOD);
    while(1){
        os.WaitEvent(TIMER0_EVENT);
        led1 = !led1;
        os.SetEvent(T1_TO_T2_EVENT, TASK2_ID);
    }
}

void task2(void)
{
    while(1){
        os.WaitEvent(T1_TO_T2_EVENT);
        led2 = 1;
        wait_ms(100);
        led2 = 0;
    }
}
```

around 11 KB of RAM, or about one third of the 32 KB available.

mbos is simple enough to be practical in a one- or two-task application, such as the blinking LED application shown in Listing 5, yet can be scaled to support as many as 100 tasks. Each task can support 32 different events. Timers and resources are equally scalable.

Finally, the API is simple but nonetheless extremely powerful. There are only 17 member functions to learn, including the class constructor. mbos is embodied in a standard mbed library and is used in exactly the same way. It is fully documented, and the code has solid error-checking and explicit error messages.

In short, I am extremely proud of the result. I look forward to seeing the many applications and extensions I know the mbed community will build onto the framework I have provided.

Andrew Levido (andrew.levido@gmail.com) earned a Bachelor's degree in Electrical Engineering in Sydney, Australia, in 1986. He worked for several years in R&D for power electronics and telecommunication companies before moving into management roles. Andrew has maintained a hands-on interest in electronics, particularly embedded systems, power electronics, and control theory in his scarce free time. Over the years, Andrew's job has taken him and his family to live in Hong Kong, Shanghai, and Paris. He is back home in Australia now, but maintains a keen interest in travel and experiencing other cultures.

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2012/258.

RESOURCES

ARM, "Cortex-M3 Technical Reference Manual," 2010, http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0337i/index.html.

———, "Keil Application Note 207: Porting an mbed Application to MDK-ARM," 2010, www.keil.com/appnotes/files/apnt207.pdf.

———, "ARM Development Tools, Keil MDK_ARM v4.0," 2011, www.keil.com/arm/mdk.asp.

mbed, http://mbed.org.

NXP Semiconductors, "UM10360, LPC17*xx* User Manual," 2010, ww.nxp.com/documents/user_manual/UM10360.pdf.

SOURCES

Cortex-M3 Processor
ARM Ltd. | www.arm.com

LPC1786 Microcontroller and mbed
NXP Semiconductors | www.nxp.com





Elektor Shop

The world of electronics at your fingertips!

Kits & Modules



Improved Radiation Meter

This device can be used with different sensors to measure gamma and alpha radiation. It is particularly suitable for long-term measurements and for examining weakly radioactive samples. The photodiode has a smaller sensitive area than a Geiger-Müller tube and so has a lower background count rate, which in turn means that the radiation from a small sample is easier to detect against the background. A further advantage of a semiconductor sensor is that is offers the possibility of measuring the energy of each particle, allowing a more detailed investigation of the characteristics of a sample. The optional PC-based software displays the energy spectrum, permitting a very detailed analysis to be carried out.

Kit of parts incl. display and programmed controller

Art.# 110538-71 • $57.30

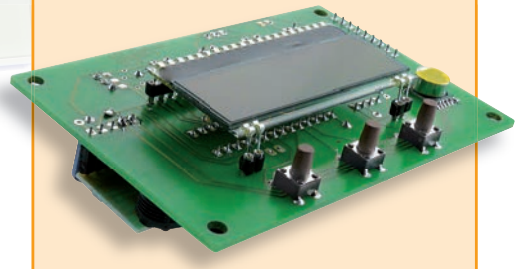


FT232R USB/ Serial Bridge/BOB

You'll be surprised first and foremost by the size of this USB/serial converter — no larger than the moulded plug on a USB cable! And you're also bound to appreciate that fact that it's practical, quick to implement, reusable, and multi-platform — and yet for all that, not too expensive! Maybe you don't think much of the various commercially-available FT232R-based modules. Too expensive, too bulky, badly designed, ... That's why this project got designed in the form of a breakout board (BOB).

PCB, assembled and tested

Art.# 110553-91 • $20.90



USB Long-Term Weather Logger

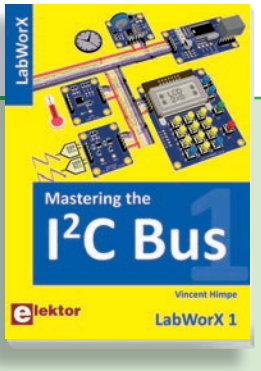
This stand-alone data logger displays pressure, temperature and humidity readings generated by I²C bus sensors on an LCD panel, and can run for six to eight weeks on three AA batteries. The stored readings can be read out over USB and plotted on a PC using gnuplot. Digital sensor modules keep the hardware simple and no calibration is required.

Kit of parts incl. PCB, controller, humidity sensor and air pressure sensor modules

Art.# 100888-73 • $50.20

Prices and item descriptions subject to change. E. & O.E

Books

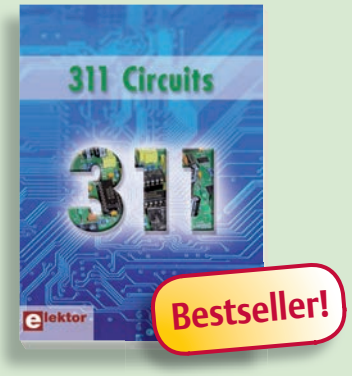


LabWorX: Straight from the Lab to your Brain

Mastering the I²C Bus

Mastering the I²C Bus is the first book in the LabWorX collection. It takes you on an exploratory journey of the I²C Bus and its applications. Besides the Bus protocol plenty of attention is given to the practical applications and designing a solid system. The most common I²C compatible chip classes are covered in detail. Two experimentation boards are available that allow for rapid prototype development. These are completed by a USB to I²C probe and a software framework to control I²C devices from your computer.

248 pages • ISBN 978-0-905705-98-9 • $47.60

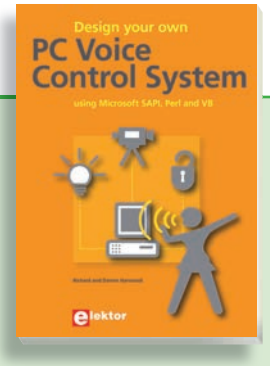


Creative solutions for all areas of electronics

311 Circuits

This book contains circuits, design ideas, tips and tricks from all areas of electronics: audio & video, computers & microcontrollers, radio, hobby & modelling, home & garden, power supplies & batteries, test & measurement, software, not forgetting a section 'miscellaneous' for everything that doesn't fit in one of the other categories. 311 Circuits offers many complete solutions as well as useful starting points for your own projects. Both categories and anything in between represent a veritable fountain of inspiration for cultivating your own ideas and learning about electronics.

420 pages • ISBN 978-1-907920-08-0 • $47.60



Talk with your computer

Design your own PC Voice Control System

This book guides you through practical speech recognition, speech annunciation and control of really useful peripherals. It details a project which will enable you to instruct your computer using your voice and get it to control electrical devices, tell you the time, check your share values, get the weather forecast, etc. and speak it all back to you in a natural human voice. If you are interested in the practical technology of interfacing with machines using voice, then this book is your guide!

216 pages • ISBN 978-1-907920-07-3 • £29.50

Elektor is more than just your favorite electronics magazine. It's your one-stop shop for Elektor Books, CDs, DVDs, Kits & Modules and much more!
www.elektor.com/shop

elektor

Elektor US
PO Box 180
Vernon, CT 06066
USA
Phone: 860-875-2199
Fax: 860-871-0411
E-mail: order@elektor.com

CD/DVD-ROMS



Circuits, ideas, tips and tricks from Elektor

CD 1001 Circuits

This CD-ROM contains more than 1000 circuits, ideas, tips and tricks from the Summer Circuits issues 2001-2010 of Elektor, supplemented with various other small projects, including all circuit diagrams, descriptions, component lists and full-sized layouts. The articles are grouped alphabetically in nine different sections: audio & video, computer & microcontroller, hobby & modelling, home & garden, high frequency, power supply, robotics, test & measurement and of course a section miscellaneous for everything that didn't fit in one of the other sections. Texts and component lists may be searched with the search function of Adobe Reader.

ISBN 978-1-907920-06-6 • $55.70



Elektor's Components Database

CD ECD 6

This CD-ROM gives you easy access to design data for over 70,000 components! The program package consists of eight databanks covering ICs, transistors, diodes and optocouplers. A further eleven applications cover the calculation of, for example, zener diode series resistors, voltage regulators, voltage dividers and AMV's. A colour band decoder is included for determining resistor and inductor values. All databank applications are fully interactive, allowing the user to add, edit and complete component data. This CD-ROM is a must-have for all Electronics enthusiasts!

ISBN 978-90-5381-258-7 • $40.20

by Steve Johnson (USA)

The SMD Air Bath

A DIY PCB Preheating Setup

The SMD Air Bath is a temperature controller that can preheat a surface-mount device printed circuit board assembly before reflow with a hot air pencil. The preheating capability activates the flux, reduces the chance of thermo shock, relieves the heat load from the hot air pencil, and reduces thermo-mechanical stresses.

Over the last 15 years surface-mount devices (SMDs) have revolutionized the electronics industry. Designs integrating these devices have benefited by smaller component sizes, higher component pin counts, lower power requirements, component placement on both sides of the printed circuit board (PCB), better mechanical performances, and excellent thermal performances. The main disadvantage of these components is they require more sophisticated production equipment that makes the manual prototyping, repair, and rework more difficult.

Entrance into this prototyping/repair world requires several tools. The development of PCB assemblies may require a reflow oven, where repair and rework requires more than just a hot air pencil. The most overlooked necessity in this repair/rework process is the preheating of the PCB.

My "SMD Air Bath" design enables the preheating of a PCB assembly prior to the reflowing with a hot air pencil (see Photo 1). This preheating provides time for the flux to activate; it decreases the temperature difference between the assembly and reflow, reducing the chance of thermo shock; it counteracts the heatsinking properties of heavy power planes, relieving the heat load from the hot air pencil; and it enables a controlled cooldown, which minimizes thermo-mechanical stresses.

The heart of the SMD Air Bath is a profiled temperature controller. This controller provides configurable ramp up, set point, and cooldown and provides pulse width modulation (PWM) for fan control. With code modifications, it can also control the heat profile of a toaster oven completing our toolbox with a home brewed reflow oven (see Photo 2). But first let's examine the air bath.

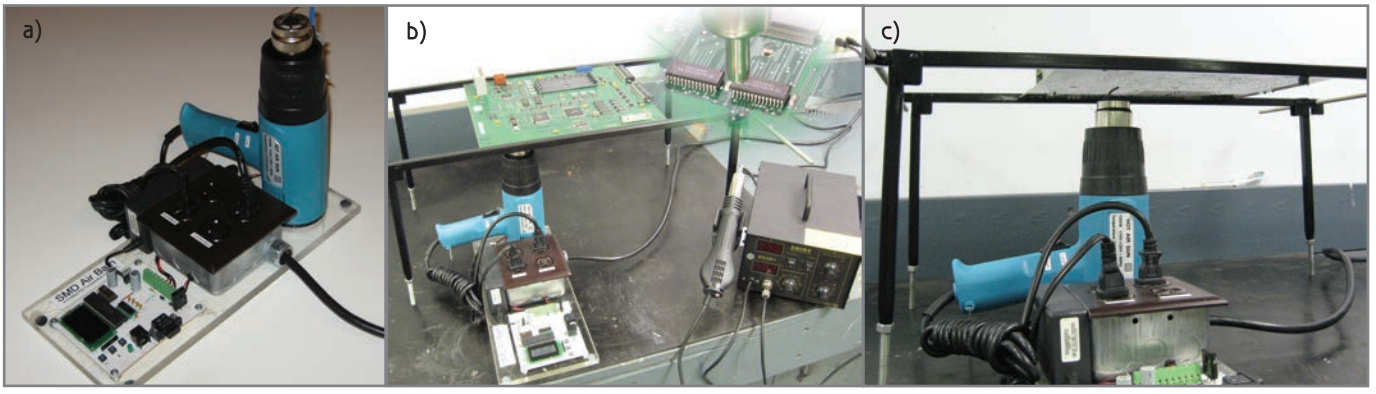


Photo 1a—The SMD Air Bath is a profiled temperature controller. **b**—The SMD Air Bath is placed underneath an SMD circuit board/component. **c**—The preheat cycle is executed, the component is reflowed, and then the cooldown cycle is entered for the replacement of the SMD component.



Photo 2—The system connected to a reflow oven

HOT AIR GUN & PROFILED CONTROLLER

The purchase of an economical hot air gun started this project. Its dissection was required to provide access to the fan's motor connections. A terminal strip and switch was added enabling normal fan operation as well as external control. A thermocouple was also attached using screw terminals enabling easy removal (see Photo 3). The modifications enabled monitoring and control, but with the quick removal of the thermocouple still provide normal use of the hot air gun.

The schematic of the controller is shown in Figure 1. The circuit is supervised by a Microchip Technology PIC18F4520 microcontroller and is pretty straightforward. The thermocouple feeds into thermocouple amplifier U2, which is an Analog Devices AD595 type K thermocouple amplifier that provides cold junction compensation and an output voltage of 10 mV/°C. This signal is fed through integrator R6 and C11 and then into the analog-to-digital convertor (ADC) of the PIC18F4520. Pin RC0 controls a solid-state relay that energizes the heating element. Pin RC2 is the PWM output that controls MOSFET Q1 providing fan control.

Push button inputs are on RB0, RB1, and RB2. LCD data and control use port D requiring the liquid crystal display (LCD) to operate in 4-bit mode. TX and RX pins connect to U3, a Maxim Integrated Products MAX232 driver/receiver, providing RS-232 communications. U4 provides the 5-V power and J3 provides access for Microchip's MPLAB ICD 2 programmer.

The controller is designed with ease of programming and RS-232 communications in mind. The ease of programming enables the system to be modified for other applications like the reflow oven described earlier. It also helps in the reprogramming of variables for the tuning process. The RS-232 communication enables tracking of the actual heat profile and is instrumental in the tuning process. The PIC18F4520 is overkill for the project, but with its use on Microchip's PICDEM 2 Plus development board, is an easy choice for the project. To aid in fabrication, a PDF of the PCB layout is available on the *Circuit Cellar* FTP site.



Photo 3—Modifications to the hot air gun enable temperature monitoring and fan control but with the removal of the thermocouple still enable normal use of the hot air gun.

CONSIDERATIONS

In developing embedded systems, there needs to be coordination between the hardware design and the software development. Sometimes programs are developed for just the test or calibration of the hardware, but software developers need to fully understand the abilities and limitations of the hardware. In this design, the use of a thermocouple and PWM for fan control introduces isolation issues. The PWM becomes one large source of noise and with the proximity of the thermocouple this gets introduced into its amplifier. Integrator R6 and C11 help damp this noise, but further steps in software will have to be considered to eliminate it.

Another consideration is the solid-state relay for heat control. Most solid-state relays use a zero-crossing detector for the engaging of their TRIACs. This controller offers no feedback for this zero-crossing detection so the controller is firing blind. To compensate, a different style of PWM for heat control will have to be implemented—one that energizes for complete half cycles rather than energizing for certain conduction angles.

The implementation of proportional-integral-derivative (PID) and its routines are another huge consideration. These routines need to work within the hardware constraints including the use of integers, limiting set-point errors, preventing value overflows, taxing interrupts, and even controlling an asymmetric load. All these considerations affect the construction of this project. The advantage of using an embedded system is that it can be tailored for the exact control specifications needed.

PID REVIEW

Figure 2 shows a PID control loop. The control loop starts by comparing the set point with the actual process variable. This comparison results in an error indicating how far the process is from its desired location. This error is passed to three controls: the proportional control $Kp\ e(t)$, the integral control $Ki \int e(t)dt$, and the derivative control $Kd\ de(t)/dt$. The proportional control makes changes to the process based on a proportion or gain of this error. A large gain results in a larger/faster process change but can result in an unstable system; too little gain and the system doesn't respond to disturbances. In our heating application, heat losses produce an inherent gain to the process. This inherent gain under just proportional control will result in a steady-state error that drops below the set point.

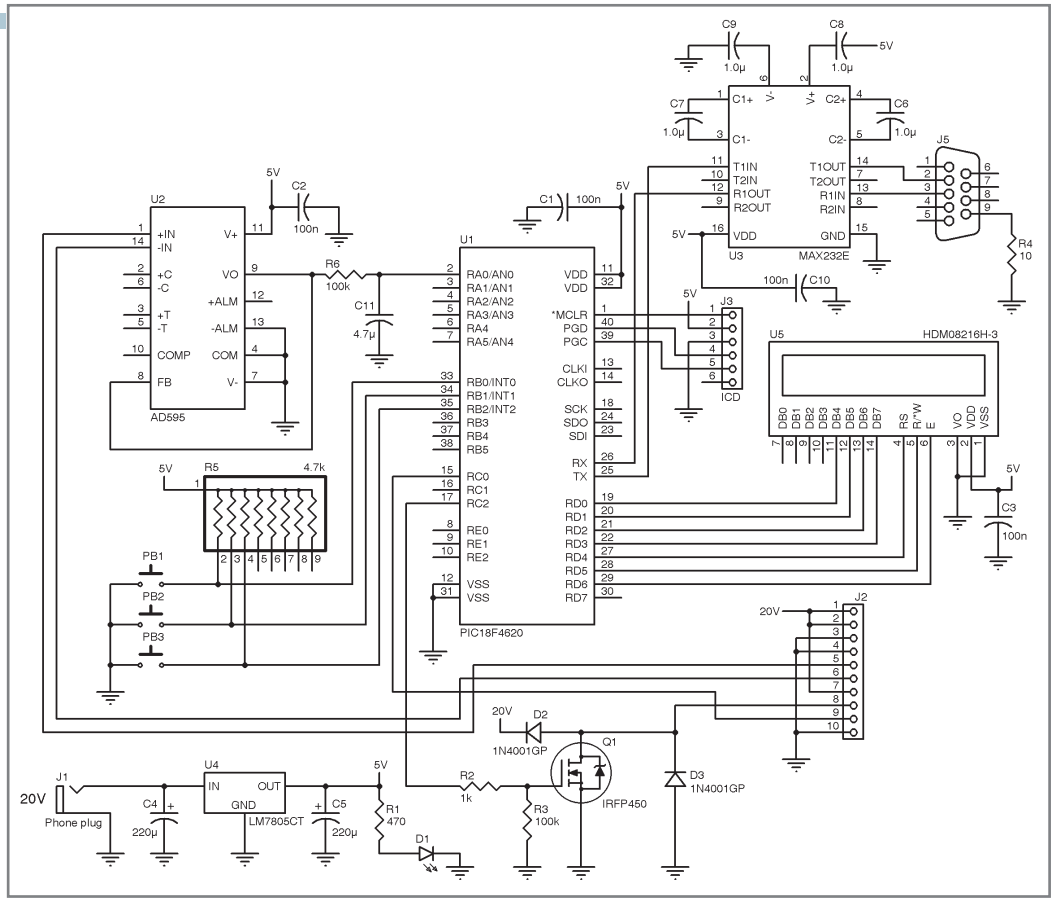


Figure 1—The SMD Air Bath is designed around a Microchip PIC18F4520 microcontroller. The system uses PWM for fan control, ADC for temperature measurement, and controls the hot air gun's heating element through a profiled algorithm. The circuit provides RS-232 communications as well as an ICD connection for ease of programming.

This droop is an equilibrium point determined by both the proportional gain and the inherent gain and may be better understood by considering that at the set point the proportional term produces no heat, but some heating is required to overcome the losses of the process, thus the system settles at a point lower than that of its set point.

The integral term compensates for these losses. It integrates or creates an average of the error over time and adds this correction term back into the process. This compensation behaves like an auto correction signal constantly adjusting for an offset in the error. A problem that arises from this correction is a condition referred to as "integral wind up" where extended disturbances end up forcing the process into an overshoot. This overshoot is due to the integral not knowing the difference between a disturbance and an error, but is based mainly on the time duration of the disturbance.

The derivative term can help limit this condition. The derivative makes changes to the process based on the slope of the changing error. The more change, the more the derivative influences the process. This helps adjust the offset quickly, limiting the time duration of the integral. As we see all terms affect each other, yet all terms need freedom to respond. It's the constraints of the system that limit this freedom such that sometimes modifications need to be implemented to better model the process. These modifications may include set-point ramping, derivative of output, freezing the integral in case of disturbances, and even replacing functions based on better modeling of the process. In order to provide the best system performance, the PID needs to be tuned and modifications that best model the desired response of the system need to be established.

THERMOCOUPLE CALIBRATION

The thermocouple was actually a piece of thermocouple wire. I took a 3" piece, stripped an inch off its end, and wrapped it together. Its contact wasn't welded, but I thought as long as its output was consistent, it would work nicely. The AD595 provides amplification, cold junction compensation, and an output voltage of 10 mV/°C for this thermocouple. Its output is read through the low-priority interrupt that gets triggered upon convergence of the ADC. These readings generate a running average that helps eliminate noise from the PWM while still correlating to an actual temperature.

To accurately calibrate this correlation, the controller needs to be tuned, but to tune the controller the correlation

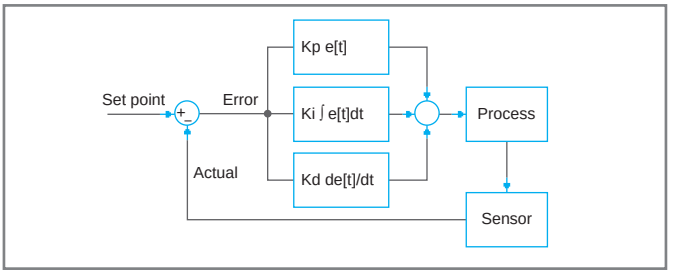


Figure 2—The PID control loop is the summation of proportional, integral, and derivative error terms where the best system response is the result of a finely tuned system.

Listing 1—The low-priority interrupt is used for the reading of the thermocouple. A weighted average is preformed to smooth and eliminate noise. The ADC value can be passed directly as the current temperature for calibration. Integer math is used to prevent the saving of the MATH_DATA section.

```
//*****************************************************************************************************************
//                                                                                                                *
// Low Interrupt Handler                                                                                          *
//                                                                                                                *
// The low interrupt reads the A/D converter.                                                                     *
//                                                                                                                *
// This routine happens approximately every 5ms. The temperature is used                                          *
// every 0.1 second so 20 samples can be read and a running average obtained                                      *
// before the temperature is needed.                                                                              *
// The use of the (3/4) (1/4) averager will reach 99.58% in 20 terms                                              *
//                                                                                                                *
// With this routine executing every 5ms the goal is to not have to save a                                        *
// lot of sections. If we use integer math we don't need to save MATH_DATA.                                       *
//                                                                                                                *
//*****************************************************************************************************************
#pragma tmpdata ISRLow_tmpdata          //Create a new tmpdata section for this interrupt
#pragma interruptlow LowIntHandler nosave = section(".tmpdata"),section("MATH_DATA")
#pragma code // Return to the default code section
void LowIntHandler(void)
{
    if(PIR1bits.ADIF)
        {
        // Average the temperature readings
        // CurrentADC = (3/4)*CurrentADC + (1/4)*ReadADC();
        // Keep it integer math
        CurrentADC = ((CurrentADC*3)>>2)+(ReadADC()>>2);

        // Remark out after calibration of the thermocouple
//      CurrentTemperature=CurrentADC;
        // Calculate the temperature using our equation
        // y = 0.6718x - 17.376
        // .6718 = 43/64
        // Keep it integer math
        CurrentTemperature = ((CurrentADC*43)>>6)-17;

        PIR1bits.ADIF           = 0;            // Reset Interrupt flag
        }
}
//******************* End Low Interrupt Handler ************************************************************
#pragma tmpdata         // Go back to the default .tmpdata
```

has to be established. It's kind of a dilemma, but if we roughly calibrate the thermocouple, the controller can be tuned, then we can better calibrate the thermocouple.

For this rough calibration, we need two thermal points with corresponding ADC values. One easy one is room temperature and another is whatever a hair dryer provides. To get the ADC values, the ADC interrupt needs to pass the ADC value back as if it's the current temperature. This gives us the ADC value on the configuration entrance screen, and we don't want to run the system as it's not tuned.

When my system was turned on, room temperature (26°C) produced a value of 64. The hair dryer produced 47°C with an ADC value of 97 resulting in an equation of $y = 0.6363 \times -23.14$. Using this equation enabled tuning of the controller, but once tuned, a better calibration was performed by setting the set point to 70 then stepping from 70 to 80, all the way up to 200, in steps of 10 to produce a thermocouple profile. This data was plotted in Excel and produced a calibration equation (see Figure 3).

Keeping the correlation conversion integer based for both equations required a little manipulation, but was required as not to bog down the low-priority interrupt with the use of floating point. This calibration method enables almost any type of linear temperature sensor to be used. It also compensates for losses/gains in the connections and fudges in some of the noise of the PWM. We also start to see the advantages of ease of programming as we tweak this calibration for the finest results.

SOFTWARE DEVELOPMENT

The firmware was developed in Microchip's MPLAB IDE interface version 8.56 with Microchip's "C" plug-in C18 version 3.36. (The complete code listing is available on the *Circuit Cellar* FTP site.) The project requires the modification and recompiling of the system LCD routines. This rebuild is to accommodate hardware pin reassignments for the LCD, where the XLCD.h file needs to reflect the use of Port D. Once XLCD.h is replaced, the device library can be rebuilt by either the make_one_device_t.bat or make_one_device_e.bat batch file, depending on the required programming mode.

The software development is separated into module files that handle elements of the hardware. External routines were created for data entry, the LCD, and the hardware setup. The main routines incorporate these routines and provide routines for the interrupts. The system generates

Listing 2—Timers 0 and 1 use the high-priority interrupt. Timer0 is configured for interrupts every 100 ms and provides the profile process based on the machine state. Timer1 provides a 5-ms execution that controls the heating element providing a duty-cycle control over half cycles.

```
//*****************************************************************************************************************
//                                                                                                                 *
// High Interrupt Handler                                                                                          *
//                                                                                                                 *
// The high interrupt will be used for the Timer0 and Timer1.                                                      *
//                                                                                                                 *
// Timer0 will happen every 100ms                                                                                  *
// Timer1 will happen every 5ms                                                                                    *
//                                                                                                                 *
//*****************************************************************************************************************
#pragma tmpdata ISRHigh_tmpdata                    //Create a new tmpdata section for interrupt and routines
#pragma interrupt HighIntHandler nosave=section(".tmpdata")
#pragma code                                       // Return to the default code section
void HighIntHandler(void)
{
//** Timer 0 for the profile machine ********************************************************************************
        if(INTCONbits.TMR0IF)                      // Timer 0
              {
              WriteTimer0(T0_DelayValue);          // Reset Timer for 100ms delay
              // Clock
              Time.Tenths++;                       // Adjust clock this routine happens ever 1/10 sec
              if(Time.Tenths>=10)                  // Reached a second
                  {
                  SerialSend=TRUE;                 // Every second to send serial data
                  Time.Tenths=0;                   // Reset the tenths
                  Time.Seconds++;                  // Adjust seconds
                  if(Time.Seconds>=60)             // Have we hit a minute
                      {
                      Time.Seconds=0;              // Reset seconds
                      Time.Minutes++;              // Adjust minutes
                      }
                  }
              // End Clock

              (*Process[State])();                 // Run the process for the machine state

              INTCONbits.TMR0IF=0;                 // Reset Interrupt flag
              }   // End Timer 0

//** Timer 1 interrupt for Heater Control ***************************************************************************
        if(PIR1bits.TMR1IF)                        // This happens every 5ms
              { // Timer 1
              WriteTimer1(T1_DelayValue);          // Reset Timer for 5ms delay
              ConvertADC();                        // Start the A/D

              // Heater control
              // HeaterPower values:
              // 0-100 then PWM control that %
              // <= 0 then OFF
              // >= 100 then PWM 100% (ON)
              // This interrupt happens every 5ms * 100 = .5 sec
              // this equates to 30 cycles or 60 half cycles.
              if(HeaterEnabled)
                  {
                  if(Timer1Counter++ < HeaterPower)
                      {
                      Heater = ON;
                      }
                  else
                      {
                      Heater = OFF;
                      }
                  }
              else
                  Heater = OFF;

              if(Timer1Counter>=100)
                  Timer1Counter=0;

              PIR1bits.TMR1IF=0;                   // Reset interrupt flag
              } // End Timer 1
}
//******************* End High Interrupt Handler ************************************************************
```

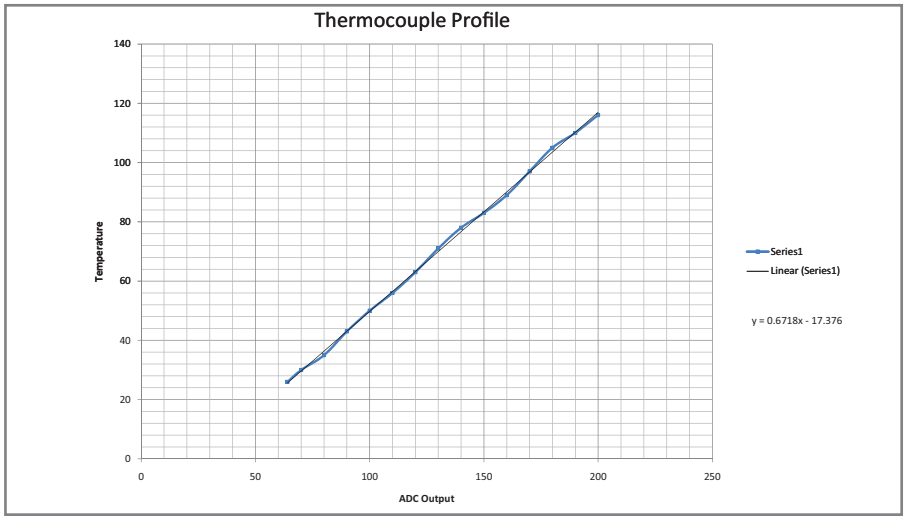


Figure 3—Once the controller is tuned, it can be stepped through different set points to determine the thermocouple profile and establish a correlation equation.

three different interrupt conditions using both the high and low priority. The low-priority interrupt is triggered every time the ADC converges. This interrupt checks that it is called by the ADC interrupt flag (see Listing 1).

Listing 3—With scope of the true hardware considerations, the routine can better model the system. The error is applied to each term: proportional, integral, and derivative. The proportional is a simple multiplication, the integral is limited, and the derivative is modified to better model the asymmetric load.

```
//*************************************************************************
//                                                                        *
// PID routine                                                            *
//                                                                        *
// Calculate the PID                                                      *
//                                                                        *
//*************************************************************************
void PID_Control(void)
{
     // Calculate the error
     Error = SetPoint - CurrentTemperature; // Calculate the error
                                            // If Error > 0 then low
                                            // If Error = 0 then @ SetPoint
                                            // If Error < 0 then high
     // Calculate the proportional value
     P=(PID_GAIN_KP*Error);

     // INTEGRAL_RESET_RATE controls the summation time
     //
     // The Error is limited to +/- 1
     if(IntegralSteps++>INTEGRAL_RESET_RATE)
         {
         if(Error>0 && Integral<INTEGRAL_ERROR_LIMIT_MAX)
             Integral++;
         if(Error<0 && Integral>INTEGRAL_ERROR_LIMIT_MIN)
             Integral--;
         IntegralSteps=0;
         }

     I=(PID_GAIN_KI*Integral);

//   Derivative = Error-PrevError;          // Calculate the derivative
     Derivative = SetPoint/CurrentTemperature; // Replacement of derivative

     // Calculate the D value
     D=(PID_GAIN_KD*Derivative);

     // Calculate the PID
     HeaterPower = P + I + D;               // Add them together
// PrevError = Error;                      // Keep the error for next time
}
//******************* End PID *********************************************
```

BEST SCOPE SELECTION & lowest prices!

25MHz Scope
Remarkable low cost 25MHz 2 channel plus trigger USB bench scope with 8 inch full color LCD display. Spectrum analysis and autoscale functions.
PDS5022S $287

iPhone Scope
5MHz mixed signal oscilloscope adapter for the iPhone, iPad and iPod Touch! A FREE iMSO-104 app is available for download from Apple App Store.
iMSO-104 $297.99

100MHz Scope
High-end 100MHz 1GSa/s 2-channel benchscope with 1MSa memory & USB memory port. Comes with a FREE scope carry case!
DS1052E $399

60MHz Scope
60MHz 2 channel digital scope with a 500 MSa/s sample rate, 10MSa memory & 8" color TFT-LCD screen.
SDS6062 $399

World's Smallest
The world's smallest MSO! This DIP-sized 200kHz 2 channel scope includes a spectrum analyzer and arbitrary waveform generator. It measures only 1 x 1.6 inches in size!
Xprotolab $49

100MHz MSO
2 channel 100MSa/s oscilloscope and 8-ch logic analyzer USB 2.0 and 4M samples storage per channel with sophisticated triggering and math functions.
DS328A $1359

Handheld 20MHz
Fast, accurate handheld 20MHz 1-ch oscilloscope.
- 100 M/S sample rate
- 3.5 in. color TFT-LCD
- 6 hour battery life
Inc. rugged impact-resistant case
HDS1021M $299.95

More selections at: www.saelig.com
Follow us on:

Saelig .com unique electronics

Listing 4—The RunMode() routine provides control over the profiled process. To provide a step test for the tuning process the system is slammed directly into the Soak mode. This is done by uncommenting State=Soak.

```
//*****************************************************************************************************************
//                                                                                                                 *
// Run Mode                                                                                                        *
//                                                                                                                 *
//*****************************************************************************************************************
void RunMode(void)
{
char string[50];

    SetDCPWM1((unsigned int) 6.25*FanDutyCycle);          // Set the duty cycle
    RomDisplayLCD(    "Settling",
                      " Temp");

    Delay10KTCYx(250);                                  // Fan is started which will
    Delay10KTCYx(250);                                  // affect temp with noise
                                                        // this delay will allow it
                                                        // to be averaged in
    InitRunVars();

    putrsUSART ("\n\rAirBath\n\r\n\r");

    HeaterEnabled=1;
    State=Preheat;

    // Force into Soak mode for step response
    // State=Soak;
    do
        {
        DisplayTemperature(State,SetPoint,CurrentTemperature);
        DisplayTime(&Time);

        if(State==Soak || State==Preheat)         // Allows us to change temp when in Soak/Preheat mode
                {
                if(UpButton && SoakTemp<SYSTEM_MAX_TEMP)      // Increase Temp
                {
            SoakTemp++;
            Delay10KTCYx(30);
            }

        if(DownButton && SoakTemp>SOAK_TEMP_MIN)              // Decrease Temp
            {
            SoakTemp--;
            Delay10KTCYx(30);
            }
        }
    if(EnterButton && State!=Cool)                            // Enter Cooldown mode
        {
        if(SetPoint<CurrentTemperature)
            CoolFromTemp=SetPoint;
        else
            CoolFromTemp=CurrentTemperature;
        State=Cool;
        }

    if(SerialSend)
        {
        sprintf(string,"%d\n\r",CurrentTemperature);
        putsUSART(string);                              // echo out temperature
        SerialSend=FALSE;
        }
    }while(State!=Done);
// We're done, unit has cooled

State=Init;           //Reset the process to exit
HeaterEnabled=0;
InitRunVars();

RomDisplayLCD( "Process",
"Complete");

Delay10KTCYx(250);
SetDCPWM1(0);   // Turn off Fan
}
//******************* End RunMode ***************************************************************************
```

It then reads the ADC value, divides it by four, and adds it to three-fourths of the running average.

This method of averaging will reach 99% of its true value within 20 samples and is an easy integer method to implement. This value (`CurrentADC`) is then converted through integer math to its corresponding temperature. It's this line of code that would be commented out for the calibration of the thermocouple and `CurrentTemperature=Current ADC` would be uncommented. The final step in this interrupt is to reset the ADC interrupt flag preparing it for the next ADC trigger. This interrupt routine needs to be quick. The importance of keeping all math operations integer-based enables us to not save the MATH_DATA section. And with the creation of a new section `#pragma tmpdata ISRLow_tmpdata`, we can throw away the `.tmpdata` section too. These nosaves conserve on overhead from both saving and restoring these memory sections each time the interrupt is executed.

The high-priority interrupt is triggered by both Timer0 and Timer1. The use of two timers enables ease of changing delay values and, from a programming point, provides routine separation based on interrupt flags. Timer0 provides a 100-ms (1/10-s) time base for the profile machine. This profile machine has five machine states: `Init` (initialize), `Preheat`, `Soak`, `Cool` (cool-down), and `Done`. Routines are created for each state and an array `(Process[])` points to each of them.

Upon checking for Timer0, the Timer0 value is reloaded, a time structure is adjusted for tracking minutes and seconds, and the process is called based on the machine state (see Listing 2). The final command in the routine resets Timer0's interrupt flag. This routine is the main time standard for the program. It provides a time base for all temperature ramps and handles all machine states.

Originally, I played around with the resolution of Timer1 for the heater control, realizing that a half cycle of the AC line takes 8.3 ms and the controller is firing blindly. I settled on 5 ms as it produced a semi-smooth response with a few dead spots. The Timer1 code section reloads Timer1's delay value, starts the ADC, then, if the heater is enabled, turns it on or off depending on the comparison between the HeaterPower and a counter that's incremented within the routine. When this counter reaches 100, it is reset back to 0 and we finish by resetting Timer1's interrupt flag (see Listing 2). The Timer1 routine boils down to a duty cycle (0–100) stored in HeaterPower that controls 60 half cycles where a value of 50 will energize 30 out of 60 half cycles producing a 50% duty cycle. This routine handles the concerns of the solid-state relay where it is unable to energize for certain conduction angles but as we see 100 controlling 60. We've created quite a few dead spots.

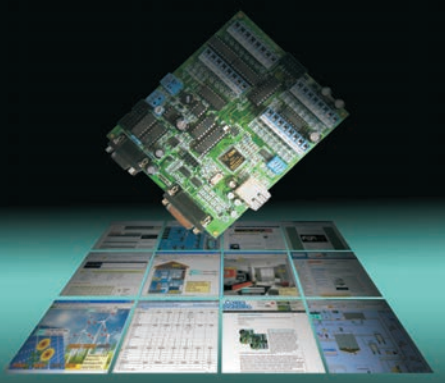
Implementing a PID control in an integer environment is a bit tricky. I've always been told that the relationship between theory and practice is a theoretical one, and I have to remind myself that I'm not programming in the real (number) world here. Listing 3 shows the comparison between the set point and actual temperature.

This comparison produces the error signal that is common to all three control terms. The proportional control is straightforward. It just multiplies the error by `PID_GAIN_Kp` (the proportional gain). The integral term is where integers can introduce a huge problem. In looking at the worst case, the heat gun is capable of producing temperatures in excess of 375°C. Depending on the set point, this could produce errors in the hundreds. Summating these errors at 1/10 of a second will quickly result in a large number. The heater control requires a small number (0–100) so this integral term will be overwhelming. But, if we limit the error for the integral term and control the summation time, we can bring the integral term back into the controllable region of the output. The advantage of this is it can self adjust across the full process control. And tuning is more of an enabling rather than finding the correct gain. The disadvantage is we lose some of the true characteristics of the integral.

The derivative is the calculation of the error minus the previous error. This can be seen commented out in

INTEGRATED

Super PLCs with Integrated Ethernet



Built-in :
- **Ethernet / MODBUS TCPIP**
- **Digital & Analog I/Os**
- **Serial ports**
- **PWM/PID/Stepper Control**

Integrate your OEM machines into the world of of Internet-connected devices with Super PLCs from Triangle Research International. The open-board style and advanced features of the Ethernet-equipped Super PLCs make them ideal for embedding into OEM applications.

Our acclaimed TRiLOGI Ladder+BASIC software reduces complex task programming to a few simple keywords - to date, over half a million copies of TRiLOGI have already been downloaded by enthusiastic users worldwide. Be a part of our international network.

tel : 1 877 TRI-PLCS
fax : 1 775 254-1816
web : www.tri-plc.com/cci.htm

TRIANGLE RESEARCH INTERNATIONAL

Listing 3. As we discussed about the derivative, the more change, the more it influences the process. This means that once the actual temperature is close to the set point the derivative has little to no effect on the system. So, then the derivative affects the system when it's over temperature returning to set point and when it's under temperature approaching set point. If we analyze this heating system, it has an inherent heat loss. Heat is constantly being sucked into the environment. The heater is also an asymmetric load, meaning that if we doubled the duty cycle we wouldn't double the output temperature. With this understanding we're less worried about the condition of being over temperature returning to set point than we are about under temperature obtaining set point. The replacement of the derivative term with the equation SetPoint/CurrentTemperature provides just that result. When CurrentTemperature is below SetPoint, a large value is produced; when CurrentTemperature equals SetPoint, it produces 1; and when CurrentTemperature is above SetPoint, it produces 0 (remember integer math). This control may not be a true theoretical PID control, but it's modeled after the intentions of PID and makes scope of the true hardware considerations.

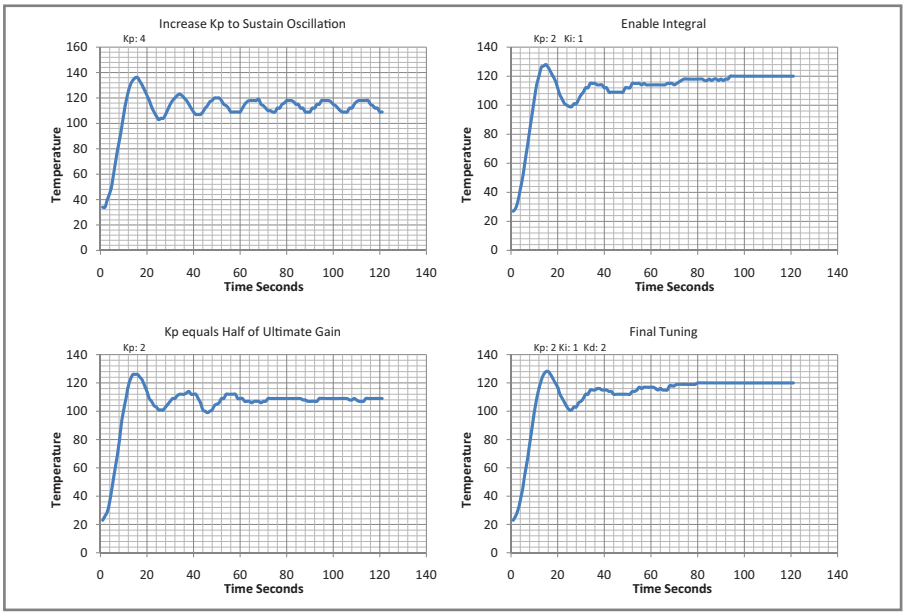


Figure 4—Each graph shows the progression of the tuning process. The upper left is the result of increasing Kp to the point of oscillation, called ultimate gain. The lower left is using half of this ultimate gain. The upper right is with the integral enabled and the lower right is the final tuning.

TUNING

An ideal system would track set-point changes to an exact process control. Real-world situations yield responses that are either under, over, or critically damped where each situation resigns to a particular preferred performance. This performance is coupled to the tuning whose intention is to provide a stable response that best maintains the thresholds of the system. There are many tuning techniques for a PID controller. Examples include Ziegler-Nichols, Tyreus-Luyben, Cohen-Coon, and Lambda, but all provide ballpark figures. In these cases, the final tuning is still done by trial and error.

For our controller, we'll use methods described by Ziegler-Nichols but manually tune the integral and derivative. For our tuning process use the following steps: Step 1, calibrate the thermocouple using hair dryer as mentioned earlier. Step 2, set the machine state `State=Soak` in the `RunMode()` routine. Step 3, set integral and derivative gains to 0. Step 4, set proportional gain to 1. Step 5, perform a step test and plot the response over a two- or three-minute interval. Step 6, increase proportional gain (2, 3, 4,…), repeating Step 5 until the response sustains oscillations. Step 7, reset the proportion gain to roughly half of this ultimate gain. Step 8, set integral gain to 1. Step 9, perform a step test and plot the response over a two-minute interval. Step 10, set derivative gain to 1. Step 11, perform a step test and plot the response over a two-minute interval. Step 12, increase derivative (2, 3, 4,…), repeating Step 11 until the best response is observed. Step 13, the system is now tuned, remove `State=Soak` in the `RunMode()` routine. Step 14, recalibrate the thermocouple. Step 15, adjust `INTEGRAL_RESET_RATE` as needed.

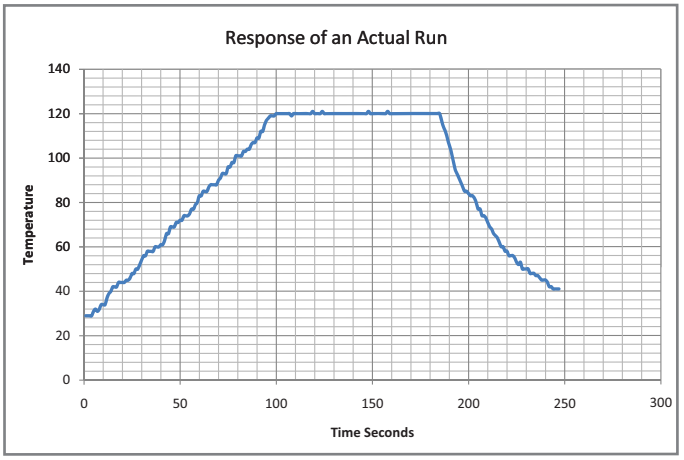


Figure 5—The SMD Air Bath will control ramp up, soak temperature, and cooldown all based on configuration settings.

As we can see in the list of steps above, the controller is going to be subjected to several step tests. The step test is performed by establishing a set point of 120°C then forcing the controller directly into the Soak mode. This is done by the uncommenting of `State=Soak` in the `RunMode()` routine (see Listing 4).

The process is then executed, RS-232 data is collected for an interval long enough to show a good

response curve, and the center (Enter) button is pressed to provide a controlled cooldown for the hot air gun. The RS-232 data is plotted and the response is used to determine the next parameter change (see Figure 4). Enabling the integral term is simple. A value of 1 performs this and enables the term to self adjust across the full process control. The INTEGRAL_RESET_RATE may need to be adjusted, but this can be accomplished well after the gain tuning is established. The derivative is established completely by trial and error. Plotting of the RS-232 data enables easy interpretation of modifications, parameter changes, and even remodeling of a process. If a code section doesn't perform, tailor it for the desired result, as this project is all about ease of programming and communications.

OPERATION

With all the calibration and tuning done, the operation of the system is pretty simple. Pressing the down arrow enables configuration of the system. This configuration sets the fan duty cycle, heat rate, soak temperature, and cooldown rate. The heat and cooldown rates are multiplied by 10, so a value of 15 would relate to a 1.5°C/s heat rate.

Once configured, the Enter button runs the process. Upon execution, the system will ramp up, maintain temperature with an option to increase or decrease based on the pressing of the up or down arrows, then cooldown when the Enter button is pressed. A run of the system produces a response something like what is shown in Figure 5.

The actual, true operation of the system is now dependent on the solder paste. The solder paste parameters are used to configure the air bath. In my case, a Sn63/Pb37 paste required a heat rate of 1.5°C/s, soak temperature of 150°C, and a cooldown rate of 2.0°C/s. Upon execution, remember the dwell time and enable the flux to activate. In my case a 60-s dwell time showed a softened paste that started to flow into the pads. The hot air pencil was used to reflow the pins and, upon completion, the cooldown finished the SMD replacement.

This project didn't reflect the creation of a rack for holding the circuit board, but almost anything can be constructed remembering that the SMD Air Bath gets hot. The trick of using the system is adding just the right amount of solder paste. Too much, and the pins become one large jumper. Not enough, and the pins aren't secured.

PROTOTYPING & REPAIR

In this article, I described the development of an embedded system: the circuit design, the software creation, all the fine tuning, and its actual usage. I purposely highlighted the connection between software and hardware development, and I emphasized specific capabilities and limitations. You're now prepared for SMD prototyping and repair.

Steve Johnson (Steve_Johnson@ferris.edu) is an electronics technician at Ferris State University. He has a Bachelor of Science degree in Electrical/Electronic Engineering Technology, a Computer Science minor, and an Associates degree in Industrial Electronics, with continuing education toward a Pre-Engineering degree. Steve enjoys building electronic projects, printed circuit boards, and developing embedded systems.

PROJECT FILES

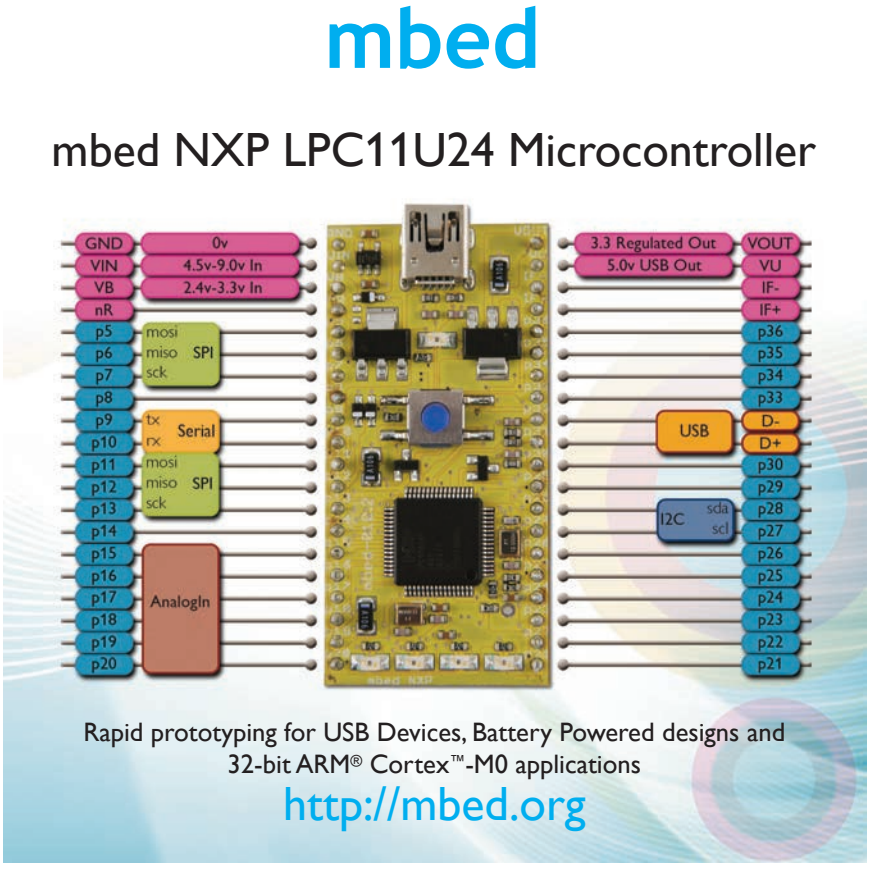
To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2012/258.

SOURCES

AD595 Type K thermocouple amplifier with cold junction compensation
Analog Devices, Inc. | www.analog.com

MAX232 Driver/receiver
Maxim Integrated Products | www.maxim-ic.com

PIC18F4520 Microcontroller, PICDEM 2 Plus development board, and MPLAB ICD 2 programmer
Microchip Technology, Inc. | www.microchip.com



QUESTIONS & ANSWERS

Innovative Research & Design

An Interview with Mike Smith



Mike Smith is a professor of Computer Engineering at the Schulich School of Engineering, University of Calgary, Canada. His main interests are in developing new biomedical engineering algorithms and moving them onto multicore and multiple-processor embedded systems in a systematic and reliable fashion. Mike recently became a convert to the application of Agile Methodologies in the embedded environment. Mention "test-driven development" and his eyes light up. In 2011, Mike had his Analog Devices University Ambassadorship renewed for the eleventh straight year. In December, Mike and I discussed his interests and some of his students' projects.—*Nan Price, Associate Editor*

NAN: Where in Canada do you live?

MIKE: I live in Calgary, Alberta, near the Rockies and home of the Calgary Stampede that is celebrating its 100th birthday in 2012. Believe it or not, although 1,000 km from the Pacific and 3,000 km from the Great Lakes, Calgary is a hotbed of sailing, both when the water is soft (summer) and when it is hard (winter). My interest in sailing stretches to racing Fireballs—a 16′ two-man dinghy that can be a lot of fun to handle in strong winds. I also volunteer with the Disabled Sailing Association of Alberta (DSAA), which works on ways individuals and groups with various levels of physical ability can sail and race, independently.

NAN: Take us back in time. How did you get started with electronics?

MIKE: Electronics when I was at school meant high school physics working with resistor boxes (called post office boxes in the U.K.), ballistic galvanometers, LaClanche wet cells, and open pools of mercury for good electrical conductivity. The latter is probably the reason why I don't remember much about that period of time.

My high school physics teacher was the one who inspired me with a real interest in doing hands-on stuff, but when did I start electronics? It's difficult to say. I suppose my dad did show me how to replace the fuse wire (not the fuse) in the household mains box without electrocuting myself. Does that count? I also took apart and fixed a Victorian shock machine that was used at 19th-century parties to transmit small electrical shocks to long lines of party goers who were holding hands. It was basically a vibrating spring to make and break an inductive circuit. Anybody using that today would probably be sued!

NAN: What did you study in college?

MIKE: I studied physics and mathematics at the undergraduate level and solid-state physics at the graduate level. At the graduate level, I got introduced to nuclear magnetic resonance of solid-state materials. The algorithms I started to learn there later turned into the MRI biomedical engineering research I am still doing. A neat part of my thesis was "mechanical signal processing," where I designed and used an air turbine to spin my magnetic resonance samples at 4 kHz. Sounds more impressive expressed as 240,000 RPM. That research was rather expensive for my supervisor, Gwyn, as I had to use helium gas as the propellant.

NAN: After completing your degrees, did you remain in academia or did you work in the electronics industry?

MIKE: After my degrees, I moved into a post-doctoral position in physics and pathology to do some early studies using magnetic resonance to identify breast cancer. In those days, it was build your own everything. I had plans to use 4004 processors and remember wiring plug-in MR ROM cards built with hand-soldered diodes.

I did a stint teaching physics, math, and computing at junior and senior high school—being the first teacher in Calgary school system to have a computer in the classroom. I founded and ran the Calgary schools' programming competition for around 20 years. I returned to academia to teach computer graphics and microprogramming when the electrical engineering department expanded into computer engineering. This was a time when a high-end graphics terminal took eight video graphics cards, each 256 × 256 × 1 bit and costing $800 a card.

NAN: How long have you been teaching at the University of Calgary?

MIKE: How long have I been teaching? I remember toggling switches in the start sequence to boot a PDP-8, how to

use a figure of eight to make sure that your paper tape reader does not jam, when a 1-K boot ROM was a card 2′ × 1′ in size, and using 1-M floppy disks that were 20″ across. I started to learn about optimized coding early since I needed overlays just to be able to run several subroutines accessing common memory locations. Convincing myself to document code was hard with a 10-character per second teletypes for a printer. I wrote my own compiler and linkers that first searched the TI980 "core" memory to find locations on the board that no longer worked. Bill was right—16 K was enough when 1 M of extension memory costs $1,400.

Now I teach embedded systems and medical-device development courses to third- and fourth-year students in computer and software engineering, with electrical engineering students taking the course as an elective. I have received support from Analog Devices to equip my teaching and research labs using the latest versions of Analog Devices microcontrollers (Blackfin and SHARC).

NAN: You're a prolific writer. Since 1992, you've published 20 articles in *Circuit Cellar*. And there are all the conference and journal papers you've published since the early 1970s. Back in the early '70s, did you make a conscious decision to become a consistent author or was it something that just sort of happened?

MIKE: I never made a conscious decision to be an author. As an academic, I have to "publish or perish" on the research side, and writing articles is a natural extension of this idea in the teaching area. I have all sorts of fun in my laboratories with my students: radio-controlled cars, one-line TVs, and the like. I always thought we could have more fun if everybody could code more efficiently. So, when something went wrong in the laboratory, I wrote up the problem in detail for my students. Then, because I was young and with a mortgage, I submitted the article. Also, if you want to get the next donation from a company to support your teaching, volunteering, or research, you have to give something back in return for the last donation they gave, and the articles also help with that.

NAN: You've written about an impressive assortment of topics ranging from programming to prototyping a voice-recognition system. While many PhDs focus on a sliver of a subject for their entire careers, someone like you seems to have technical interests that include software, hardware, design, and theory. How do you explain the diversity of your interests?

MIKE: My diversity in interests is due to the fact that I have followed Spock's advice: I have lived long and prospered in my career. I found in life that whatever is the most difficult thing to do this year is something that you will end up doing the rest of your life.

My biomedical engineering can be traced back to a need to "apply" biomedical physics concepts. I use software engineering so that my team and I can successfully complete this project and get onto the next as quickly as possible. I enjoy computer engineering because it's fun to see things really work. I'm into engineering theory because I just love throwing around symbols and playing with mathematics.

NAN: You wrote a few articles about digital signal processing and RISC chips. Explain your fascination.

MIKE: It all started with 300 bytes in a memory bank on an Apple 2. I wrote a book about how you could demonstrate ideas about how a processor worked by faking the operations of 6502 using a simulator in BASIC. I remember the book reviewer saying, "the most fascinating, frustrating, and useful part of the book was the fact that the author would first explain something. He would then follow up by getting the reader to program the concepts in a way that would deliberately fail if you typed in stuff without thinking, 'Hang on—that ain't right.'" I think my current students will recognize the technique.

However, the 6502 instruction set was ugly and not as nice and "engineeringly simplistic" as that inside the PDP-11. Then came a chance to write an article—"How RISCy is DSP?"—for *IEEE Micro* magazine. I reviewed all the current DSP, CISC, and RISC chips and speculated on a new fast processor called a "Smith chip." Supposedly, it was called "crisp" because it was hot and fast performing, but actually it was a hidden pun—chips in the U.K. are called crisps and one of the biggest manufacturers of crisps is a company called Smiths.

My fascination with matching processor architectures, DSP software, and optimizing compilers just developed from there following the creed: higher, faster, stronger. However, the basic architecture of most processors today still reflects the wire-wrapped array processors I build using byte-slice products.

NAN: In your article series "Practical Application for TDD" (*Circuit Cellar* 176–177, 2005), you and a team of designers use a test-driven development (TDD) technique for a temperature calculation project. Tell us about that project. Can you give us another example of where the TDD technique is useful?

MIKE: I can thank one of my students, Adam, and a colleague, James, at the University of Alberta for that. Adam came into my team with 15 years of industrial software development experience. He was a firm believer that business software approaches for "defect-free" code development were generally applicable. The "blue screen of death" has a special meaning when the code is running in your heart monitor. He was convinced that Agile concepts, especially "test-driven development" were a natural for the embedded world. Over the past few years, James and I have adapted testing frameworks. We proposed an embedded system, eXtreme-Programming Inspired (XPI) Agile lifecycle that works from Matlab, through C++, onto optimized C++ and Assembly code, onto on-chip coprocessors. Recently, we have been trying to adapt XPI ideas for FPGA, SoC, and ASIC devices.

Other examples of TDD and Agile concepts, eh? Examples are everywhere, since TDD approaches are just a formalization of what everybody does every day: sticking your toe into the bath water to make sure it's not too hot before plunging into it, sprinkling the baby's bottle onto your wrist before giving it to the most perfect child in the world, performing a shoulder check before you change lanes and the *key* one: making sure your boss is in a good mood before asking for a raise.

NAN: Tell us about the Microsoft Project-based nanosecond-scheduling tool you and James Miller developed ("Automatic Microcontroller Task Scheduling," *Circuit Cellar* 184, 2005). Is the tool still in use? Have you made any updates to it?

MIKE: Optimizing compilers with special DSP pragmas to match the special characteristics of your real-time code to the processor architecture are now so good that we only trot this one out at parties.

NAN: In your series "Mobile Application Development" (*Circuit Cellar* 245–246, 2010–2011), you describe how you contributed to the design of a sound-monitoring prototype used to identify medical issues related to sleep apnea. Tell us how you became involved with this project.

MIKE: Basically, two things came together: a good student wanted a summer job and I was interested in demonstrating the relationship between phone applications and what I was teaching in the embedded operating systems part of my course. Also, it was a good framework for a tool that community organizations could use to hunt down noise nuisances, such as the "Ranchlands Hum" in Calgary and found elsewhere around the world. Could you use three cell phones several kilometers apart to pick up and then transmit the 40-Hz hum to a base station for use as input to a direction finder?

Related question: Dr. Who is a U.K. time lord who travels through time and space in an embedded system called the TARDIS. He arrives in an alternative universe and goes to the local coffee shop. Everybody is typing on their cell phones. What planet is Dr. Who on?

Correct answer: Planet of the Apps. (With apologies to our local newspaper and Bizarro.)

NAN: Our editors were really excited when they learned your daughter, Lizie Dunling-Smith, contributed to your 2009 series on time-triggered systems. Working with your daughter must have been a memorable experience for you. How did that series come to fruition?

MIKE: It was really neat to work on an article with my daughter. The series came about as I need to generate detailed instructions for a fourth-year design project team on how to develop and use a cooperative scheduler in their project. Since Lizie had taken microprocessor courses, she was able to help me focus the material so I could be sure my students would understand it. Lizie has completed university and is working in the gas industry as a junior engineer—climbing up the inside of high wind-turbine towers supplying alternative energy to a northern Canadian town—leaping tall gas-stacks in a single bound.

NAN: In our recent interview with Shlomo Engelberg—a *Circuit Cellar* contributor and professor in Israel—we asked him to tell us about the topics that seem to be exciting his current group of serious engineering students. He said DSP and compressive sensing were hot topics. You too are in the trenches with the next generation of embedded designers, programmers, and electrical engineers. What hot topics are currently captivating your students?

MIKE: What topics are interesting my students? You are asking me about three weeks too early. I always have a question on my final exam that is essentially: *Marking exams is not a great deal of fun. Liven up my world—write your own exam question related to this course and answer it.* Two years ago, this question led one student, Emily, to say, "I want to use the ideas from your course to develop an embedded system that will help high school students learn how to play the oboe with the proper timbre." Emily is now my graduate student using the same analysis ideas to hunt down the Ranchland's Hum with an industrial scholarship from a local acoustics firm, Patching Associates, and the Natural Sciences and Engineering Council of Canada (NSERC).



The Snorecognizer analyzes sound-detection results and graphs information about a user's snoring behavior

So what are the hot topics? Doing real stuff and not just talking about theory. One previous ECE student, Jason Long, runs an after-school "hands-on" activity called the Micro Processor Group (MPG), which has really increased its following in the last couple of years.

NAN: Like Professor Engelberg, you use Analog Devices parts in your classes. You've been involved with the Analog University Program. Tell us about that.



A joystick or a sip-and-puff controller enables a disabled person to manage the Martin 16 Autohelm/Windlass System



A sailor controls the sails and helm of the Martin 16 sailboat using a sip-and-puff mechanism

MIKE: Basically, I used to teach using microprogrammable byte-slice chips and then 29050 RISC processors. While I was on sabbatical one year, Advanced Micro Devices switched to producing X86s. The person who taught my course did not like that idea, so he got me a class set of Analog Devices ADSP-2181 evaluation boards. When I came back from sabbatical, I found that I really liked the 2181 Assembly language syntax (R1 = R2 + R3;) for teaching more than the syntax of the Intel and Motorola chips. However, I did not think the 2181s were the way to go as I wanted single-cycle floating-point capability for my advanced DSP architecture course. So, I made a request for a donation of a class set of SHARCs (21060). That was accepted, which led to articles in *Circuit Cellar* and scientific journals.

Those articles made it easier for me to show that I really used the evaluation boards so that upgrades for teaching and research become possible. I was given the title, "Analog Devices University Ambassador" in 2001, and have been re-awarded that title each year since. The fact that local Canadian firms use Analog Devices products—and Bob Meisenhelder, the university and research grants director at Analog Devices supports my research in a variety of ways—has made it easier to gain matching grants in software, computer, and biomedical engineering from a number of industrial and government agencies to support graduate student research and undergraduate projects.

NAN: What are your main areas of interest and research at this time? What types of projects are you and your students working on?

MIKE: We are working with graduates and undergraduates to help paraplegics and quadriplegics from the DSAA sail independently using a Martin 16 sailboat with a weighted keel to keep it upright. One of my *Circuit Cellar* articles ("Voice Recognition Controlled Sailboat: Speech-Recognition Control Aids Disabled Sailors," *Circuit Cellar Online*, 1999) covered a voice controller from an earlier student project. This year, a student, Jeff, has completed a fourth-year project looking at adding a haptic controller for the Martin 16 via accelerometers attached to people's shoulders when they have limited arm movement.

One of my current embedded systems course students, Maddie, came up and said, "I would like to modify a MindWave system to manipulate a radio-controlled car with my mind." That's a pretty tough concept, but it is a great concept. So we are currently trying to raise sufficient funds (tax-deductible) to be placed in University of Calgary research account to give Maddie a scholarship and equipment for a 16-month internship. She wants to examine the practical issues of interfacing existing inexpensive "mind-control" systems to everyday objects that people with handicaps use. For example, some of the DSAA clients use a puff and sip mechanism for the DSAA Martin 16 sailboats, and we wondered if mind control was a practical alternative in a moving boat environment for somebody needing head support.

Another student, Emily, is working on combining signal-processing algorithms and a real-time embedded system to complete a low-frequency direction finder, using microphones and possible geophones, to hunt the Ranchland's Hum. The problem is the "Hum" is not just one noise nuisance but many. In some places, there is a physical hum you can record and measure, in others it is some nonlinear property inside the sufferer's head or even tinnitus, ringing in the ears. Just recently, we speculated whether a MindWave-like device could be used to identify brain waves specifically related with these noise nuisances.

Other students who are working with me include Ashley, who is trying to adapt TDD to FPGA development, and Rachel, who is doing some work on how project management needs to be customized for an ICT world. ICT for Development (ICT4D) looks like a real long-term hot topic. Some of the third-world countries are really leaping ahead in the ICT area with remote medicine, remote elections, and the like. The United Nations put out some really neat stuff (see M. Macapagal and J. Macasio's "Academy of ICT Essentials for Government Leaders, Module 7, ICT Project Management in Theory and Practice, 2009,www.unapcict.org/academy/academy-modules/english/Academy-Module7-web.pdf).

Then there are Ethan and Jason who are working on compressed-sensing algorithms to speed up magnetic resonance imaging techniques.

by George Martin (USA)



ADCs and Signal Filtering

You understand the process of creating a new product from start to finish. Now it's time to dig into the specifics, starting with analog-to-digital converters and their supporting hardware and software. Many of the techniques in this article can be applied to other design dilemmas.

In previous articles, we looked at designing a complete product. I started with requirements and ended up with detailed hardware and software designs. In between, we took a look at the step-by-step process of designing a product. We did not do much detailed hardware design. Now I will switch back to some detail work. Each of the schematics or each of the software modules in a project needs that detail type of effort. The project we just finished did not use analog-to-digital converters (ADCs). You will find that the techniques presented here can be applied to many design problems. So, let's dig into ADCs and their supporting hardware and software.

ADCs DEFINED

The ADC takes an analog signal as an input and converts that to a digital number. Consulting Wikipedia will give you a good broad-based description of ADCs. Let's start by considering only DC voltages as the input signal. Assume we are measuring the voltage of a battery and that voltage is slowly changing. This description is important to grasp and will become a basis for further exploration. A battery may discharge rapidly but with respect to the rates of conversions of modern ADCs (1×10^6 conversions per second or more), that battery looks like a DC source. Be careful in this assumption. As simple as this seems, it has problems. And those problems will lead to errors in measurements.

I'm trying to set up a conversion scenario that eliminates changes in the input signal and noise in the input signal so we can concentrate on the conversion. Any problems in the readings should be localized to the device (ADC), our interface with the device, and the power we are supplying to the device.

PARALLEL VS. SERIAL

There are two basic methods for interfacing to the converter. A parallel interface takes all the controls and puts them on dedicated lines. The status and data results are also found on separate lines. So for a 12-bit converter, we could have 15 or 20 lines between the CPU and the converter. Lots of lines and the data is transferred in one read, so it's very fast. ADCs that are internal to the CPU are interfaced in this manner.

A serial interface can be as few as two lines. One line is used for clock into the converter and another line for is used for control in and data out of the device. You are probably more likely to encounter a serial converter with a SPI (or I^2C) interface. That interface is four signals made up of clock, data_in, data_out and select. A simple converter might not have a data_in signal.

I would like to study the serial converter a bit further because it will help us understand these devices. Linear Technology has the LTC1404 ADC. My contacts at Linear will be mad that I'm not talking about the latest and greatest device. I'm not because this device is a good place to start our understanding of ADCs. Looking at the datasheet, you see that the device takes power, ground, a reference voltage, analog input, convert signal, data out, and clock.

On the input side, you need to give it power and ground and also supply a reference. Most of the converters available today are ratiometric. That means the conversion measures the relationship between the input voltage and the reference voltage. If the input is 50% of the reference then 50% of full scale is the digital output value. More modern converters have the reference voltage built in. Now, remember I said we wanted to measure a battery voltage and that should present a very stable low impedance source for conversion. Well, what happens if we have a solid input signal but noise on our reference voltage? If the reference is reduced 10%, the conversion ratio will be increased by approximately that same 10%. Even if the reference is internal to the converter, that relationship applies. Internal references have more man hours of design behind them than you will be putting into any of your designs. That means they should work well, and most do. Trust but verify. Further down the datasheet you will find graphical relationships between power supply noise and signal noise. Look at the Bipolar Mode Power Supply versus Ripple Frequency graph. V_{SS} ripple of 10 mv at 100 KHz presents –60 dB of feedthrough. And, since this device is 12 bits and the least-significant bit (LSB) represents –72 dB, we could have a problem with power supply noise feeding through. Refer to Table 1 for the value of the LSB for various converters.

A 12-bit converter has a least-significant bit (LSB) that represents 1/4,096 of full-scale reading. The decimal representation of this is not as meaningful as the dB representation. That 1 LSB in a 12-bit converter requires 72.25 dB of no other competing signal to be cleanly read. The graphs from the datasheet now have more meaning. Noise on the V_{SS} line looks like it could become a problem.

Remember this is an older part. I needed something that would show the potential problems. Modern devices from Linear and other manufacturers are much better. Also, look at a 16- or even 24-bit converter. You will need tons of power supply filtering to get a meaningful LSB.

So, in order to measure that battery, we need a reference voltage and printed circuit board (PCB) layout that will not introduce more than 72.25 dB of noise. Staying on the analog input side, let's next consider the input to the converter. A battery is a low-impedance source. But what if we were measuring a transducer or even a thermocouple? Well, we would need an input amplifier that would have enough "stiffness" so that the ADC would not change the input value as it performs the conversion. That perfect input amp is no better than the power supply powering the device. So pay attention to your buffering op-amps.

CONVERSION PROCESS

The successive approximation is a conversion method that tests each bit in the converter to see if that bit should be set or cleared. A 12-bit converter has 12 tests that take 12 clock cycles to complete. The input signal must be held constant while that conversion takes place. Most modern converters have a sample-and-hold circuit built in to perform this function. After conversion is complete, the result is available to be read out. Some serial converters start the output of the serial data as the device converts. So, reading a bit causes the next bit to be tested. This saves the throughput time so you can get the data with lower latency.

Bits	Counts	LSB	dB
8	256	0.00390625	–48.16
10	1,024	0.0009765625	–60.21
12	4,096	0.000244140625	–72.25
14	16,384	6.1035156e-005	–84.29
16	65,536	1.5258789e-005	–96.33
24	16,777,216	5.9605e-008	–144.49

Table 1—A/D resolution

WHAT DID YOU READ?

So, now you have an ADC reading. What does it mean? Well, that all depends. In the simplest situation, 0 input voltage is 0 counts and full-scale input voltage (say, 5 V) is full-scale counts (say, 4,095 for a 12-bit converter). Yes, 12 bits gives a full-scale reading of 4,095 counts so each count is 0.001221 volts per count (i.e., 5 V/4,095 counts). Isn't that ugly! We could use 5 V/4,096 counts = 0.001220703125. That's also one ugly number, and I can't say much more.

But what if your input signal was an audio signal centered above ground. Let's say the input was ±2.4 V. It would make the most sense to have the converter give you 0 counts for 0 volts input and for + full-scale give a number that was +2,047 counts for a 12-bit converter, and –2,047 counts for a –full-scale input. The A/D reading would then be a two's compliment representation. Another ugly detail is that you get +2,047 (0x7FF hex) and –2,048 (0x800 hex) counts. The best solution I've seen is to think of it as symmetrical, about 0, and assume that the lowest number is –2,047 (0x801 hex) counts. That gives a weight for each bit the same whether it's positive or negative. In our example each bit is 0.001172447484 (i.e., 2.4 V/ 2,047 counts). And that's another ugly number to say the least.

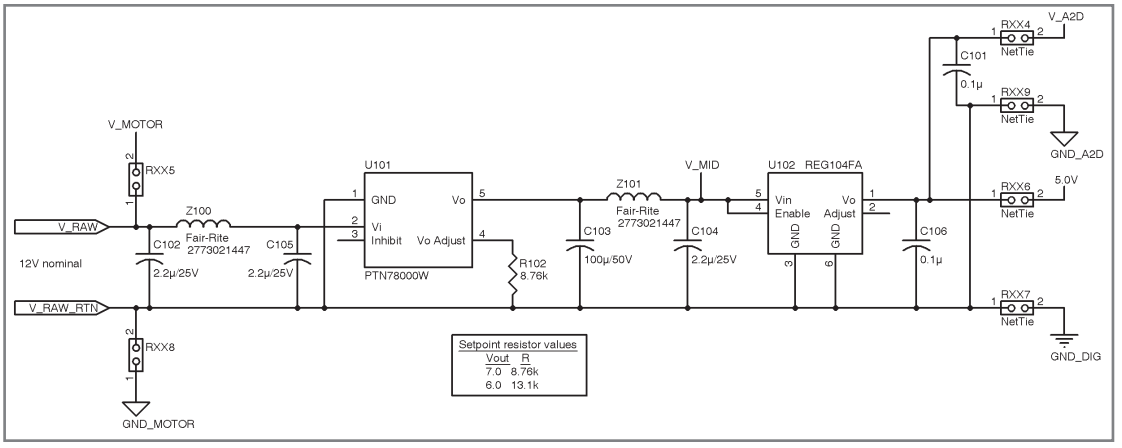
ARE WE THERE YET?

We're almost there. Let's go back to the datasheet and look at a couple more numbers. A good converter will be monotonic with no missing code. Monotonic means that for ever-increasing input signals you will get ever-increasing output codes. It does not say anything about accuracy. Imagine the converter is part of a control system. It would not be good to have the input signal increase and the conversion result decrease. That might be one definition of unstable. And no missing code means just that. All 4,096 codes are outputted in a 12-bit converter.

Next we come up to integral and differential nonlinearity. I know this stuff makes your head hurt, but it's necessary to understand for a good converter design. I encourage you to study this on your own. Use the datasheets for your particular converter. Let's just say integral and differential nonlinearity go hand in hand with monotonicity and no missing codes.

The last point is accuracy. The main component of accuracy is absolute accuracy. This is typically given as ±X LSB.

Figure 1—The power supply for a 12-bit converter



They mean to say ±X counts. And that means your reading could be off by this amount (either high or low). Funny things happen around zero (offset errors) but these are different for each converter and you need to investigate and understand your particular device. So, is this all we need to know about ADCs?

ONE LAST THING

If we are measuring battery voltage, we are probably done. If we are measuring machine vibration, there's one more topic to cover. That topic is input signal frequency and our sampling rate. This gets us into Harry's theorem. That's better known as the Nyquist-Shannon sampling theorem, or just the Nyquist Theorem. There is a lot written about this. Let me present it in a different manner. Let's assume we have a movie camera that takes pictures at 30 frames per second. Then assume we are taking pictures of a wheel on a wagon. If the wheel is moving slowly enough, every frame in our picture shows the wheel rotating ever so slightly in the proper direction (the direction of wagon travel). As the wagon moves faster, the wheel rotates more in every frame of our movie. This continues until our frame rate is the same as the rotation rate of the wheel, then the wheel doesn't appear to be rotating. In fact, just before that point in wagon speed, the wheel will appear to be rotating backward as the wagon

Innovation drives results

APEX EXPO IPC

In this economy, innovation is everything and IPC APEX EXPO 2012 is where you'll find the information to inspire it!

Discover innovative ways to manufacture, manage your supply chain and succeed in 2012. Learn at the world's premier technical conference with leading experts and top-tier educational courses. With an exhibit hall overflowing with advanced and emerging technologies, you will experience an event that is second to none.

Invest just a day or more and take home the tools that will drive your results for 2012 and beyond. Register early for FREE exhibit hall admission and extra savings at

www.IPCAPEXEXPO.org

CONFERENCE & EXHIBITION
February 28–March 1
San Diego Convention Center
2012

• DESIGN • PRINTED BOARDS • ELECTRONICS ASSEMBLY • TEST
with a new focus on printed electronics



Use your smart phone reader! Enter for a chance to win a free All-Access Registration or a free hotel stay!

INFORMATION that INSPIRES INNOVATION

moves forward. Something is happing that is distorting our measurements. And that is what happens when an input signal contains frequencies greater that the Nyquist Frequency. The input signal at the higher frequencies is folded about the Nyquist Frequency and appears as a lower frequency input. That the wagon wheel is rotating slowly backward in our movie while the wagon is moving forward is an example of the problems that arise with a sampling rate that is too low.

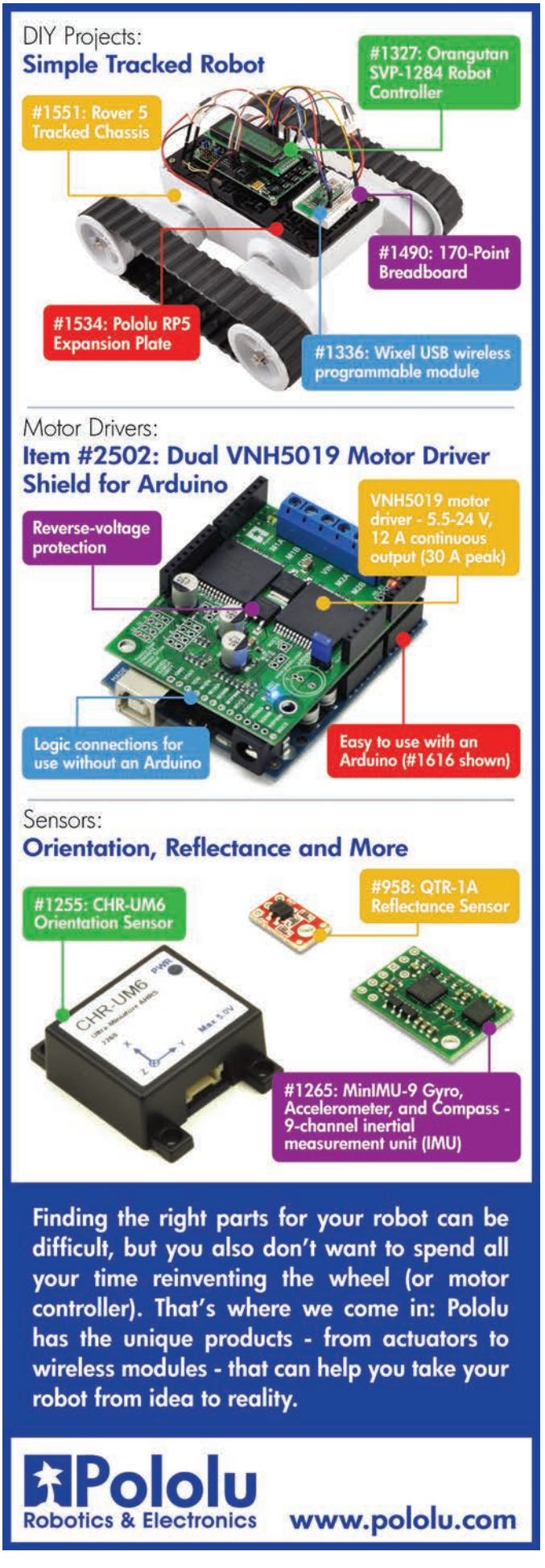
It's not a noise problem, but any frequencies too high will appear as noise folded into the lower frequencies. We need a low-pass filter to remove the higher frequencies. Read no further until you understand this point and agree with the filtering requirements. So many of my customers do not grasp this concept. It's amazing how many. These types of antialiasing filters are characterized by defining the pass area and the stop area. The transition from pass band to stop band is left loosely defined. Microchip Technology has a nice application note that deals with this filter topic. In addition to analog filters on the input, I've used digital (switched capacitor) filters on the inputs of ADCs.

NOW ARE WE READY?

Let's do an ADC design. We need a signal input. If that signal is the only signal, we can connect it to the ADC. If we have several signals we need either several converters or a multiplexer (mux) to select among the different input signals. Then we need a method to read out the ADC. Also, consider that we need power to supply the converter and a filter to keep out all the higher frequencies. Remember that we need to drive the ADC with a powerful enough signal. Also, remember a 12-bit converter needs at least 72 dB of filtering.

I selected the Linear Technology LTC1594 ADC to do our design. This device operates at 5 V, has an eight-channel input mux and a SPI output. From Linear Technology's "LTC1594 - 4-Channel, Micropower Sampling 12-Bit Serial I/O A/D Converter" webpage, you will also find the datasheet, application notes, and design notes (on the right-hand side of the page). These are valuable resources to say the least. Linear Technology is a good source to these devices as are most of the manufactures. But you need to read the fine print to make sure your selection is compatible with your requirements.

This converter is a 12-bit device that operates at 5 V. Early in the datasheet (page 2), you come across the operating conditions and characteristics. Let's look into the characteristics. We see the number of bits is 12. The input analog signal is digitized a digital output value of 0 to 4,095 counts. Next we see gain error and offset error. The gain error is the total deviation from a straight line representing the perfect conversion. For this device I find ±8 LSB (or ±8 counts). So, for any given reading, we could have an eight-count error due to A/D gain. We also find an offset error of ±3 LSB (again ±3 counts). That means near zero we could be off by three counts. Then isn't the total error possible 11 (i.e., 3 + 8) counts? Good question. You have to ask the manufacturer for a detailed answer. Typically, these errors



are not cumulative. They characterize the part's operation. So, if we asked, I bet the answer would be that these are separate. Another type of error listed is differential nonlinearity error. That error is how much each reading can deviate from the next reading as compared to the perfect converter. So a continuous error of 0.75 of a count would accumulate as we moved along the converter's operating region, but the resulting error would never be larger than the gain and offset error limits. Maxim Integrated Products has a good write up about these types of errors.

In Linear Technology's "Design Note 153: Micropower 4- and 8-Channel, 12-Bit ADCs Save Power and Space," you will find a schematic (see Figure 3 in their note) for the converter using one low-pass filter (antialiasing) for all the different signal inputs. This can be a great saving of components and costs if your design permits it. This also gives us the advantage that we only need one op-amp to drive the input to the converter. You must remember that the stronger the low-pass filter the longer you will need to wait after switching the mux input channel before you can start a conversion. If Channel 1 was near 5 V and Channel 2 was near 0 V, the low-pass filter would look at that as a big step change in the input signal and the Channel 2 signal would take some time to get through the filter. You need to add a delay whenever switching the mux position.

The last consideration for this type of converter is that you start a conversion by taking the CSADC pin low and begin clocking the SPI. The conversion is not yet complete, but the most significant bits (MSBs) are ready and can be clocked out. In fact, as you keep clocking, the other bits of the result become available and are clocked into the result.

Figure 1 is the design for the power supply for a 12-bit converter. This input power is 12 VDC. The switching regulator U101 steps that down from 12 V to a little more than 6 V. This is not a solid number in the design just yet and R102 is used to set the output voltage. Z100, C102, and C105 are an input filter for the switching power supply. There are many of these circuits connected to the 12-VDC supply, and it is in an industrial environment. Lots of motors and relays cycling. Z101 and C103 are also low-pass filters on the output of the switching supply. And finally U102 is a normal series regulator. It's one of the old Burr Brown devices (now part of Texas Instruments).

I took this approach to save power and heat loss in the system. The switcher converts at about 85% efficiency. The linear regulator can then handle the 6- to 5-V step down with not much heat generated and power lost.

Z100 and Z101 are Fair-Rite devices. They are basically inductors whose impedances vary over the input frequency and provide maximum benefit at the frequencies that we are trying to attenuate. These were selected to operate in the 100- to 155-kHz range for the switching power supply.

The components that are marked as RXXn are *net tie* components in Altium PCB software. They let you connect together nets with different names. So we can keep the wiring shown in the schematic tightly routed and the power and ground leaving for the CPU and have its own routing rules.

The original design had more filtering (in fact a completely separate regulator). But in testing at 12 bits I found that it was not necessary. So, in the schematic, I just separated the power to the CPU from the power to the ADC with simple capacitors.

NOW WE'RE DONE!

I hope you enjoyed this visit to ADC land. Notice I didn't spend much time and ink on the converter itself. As the resolution of these devices gets greater, more and more attention will have to be spent on the proper design of the entire ADC system. Design consideration such as number of bits, power supply decoupling, input filtering, and board layout will all have to be correct to get the results you expect.

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

RESOURCES

Linear Technology Corp., "LTC1404: Complete SO-8, 12-Bit, 600 ksps ADC with Shutdown," 1998, http://cds.linear.com/docs/Datasheet/1404fa.pdf.

———, "LTC1594 - 4-Channel, Micropower Sampling 12-Bit Serial I/O A/D Converter," www.linear.com/product/LTC1594.

———, "Design Note 153: Micropower 4- and 8-Channel, 12-Bit ADCs Save Power and Space," 1997, http://cds.linear.com/docs/Design%20Note/dn153f.pdf.

Maxim Integrated Products, "Application Note 2395: Understanding Integral Non-Linearity Errors," 2003, www.maxim-ic.com/app-notes/index.mvp/id/2395.

Microchip Technology, Inc., "AN699: Anti-Aliasing, Analog Filters for Data Acquisition Systems," 2002, ww1.microchip.com/downloads/en/AppNotes/00699b.pdf.

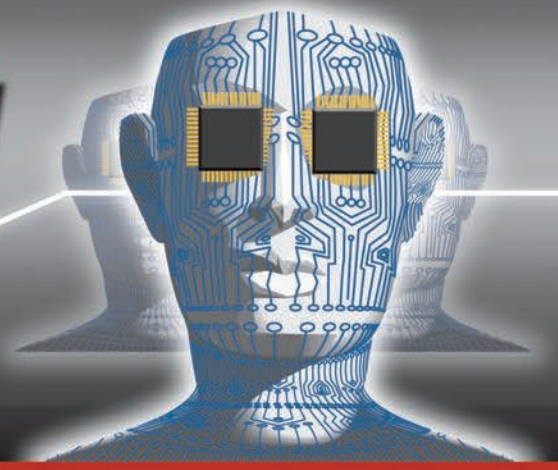
Texas Instruments, Inc., "U-102 Unitrode Application Note: UC1637/2637/3637 Switched Mode Controller for Dc Motor Drive," 1999, www.ti.com/lit/an/slua137/slua137.pdf.

SOURCE

LTC1404 and LTC1594 Analog-to-digital converters
Linear Technology Corp. | www.linear.com

DesignCon® 2012

WHERE CHIPHEADS CONNECT



Conference: January 30 - February 2
Exhibition: January 31 - February 1
Santa Clara Convention Center | www.designcon.com

Don't miss the definitive event for chip, board, and systems designers.

Join thousands of engineering professionals who make the decision to start the year off right with DesignCon!

INDUSTRY TRACKS:

- Chip-Level Design for Signal/Power Integrity
- Analog and Mixed-Signal Design and Verification
- FPGA Design and Debug
- System Co-Design: Chip/Package/Board
- PCB Materials, Processing and Characterization
- PCB Design Tools and Methodologies
- Memory and Parallel Interface Design
- High-Speed Serial Design
- High-Speed Timing, Jitter and Noise Analysis
- High-Speed Signal Processing, Equalization and Coding
- Power Integrity and Power Distribution Network Design
- Electromagnetic Compatibility and Interference
- Test and Measurement Methodology
- RF/Microwave Techniques for Signal Integrity

KEYNOTE SPEAKERS:

Tuesday, January 31



Ilan Spillanger
VP Hardware and Technology, Interactive Entertainment Business Unit, **Microsoft**

Wednesday, February 1



Prith Banerjee
SVP Research, **Hewlett Packard** *and Director of* **HP Labs**

Register today at www.designcon.com.
Use promo code ELCC to save 15% off your conference pass. Expo registration is FREE.

OFFICIAL HOST SPONSOR:

Agilent Technologies

by George Novacek (Canada)



Build a Robust Power Supply (Part 1)

Power Dissipation

Building a robust power supply design seems like an easy task. The first part of this series on power supply design focuses on switching regulator modules, power dissipation, and low-voltage operation.

Thanks to the availability of a wide range of ready-made power supply integrated circuits (ICs) and modules, power supply (PS) design seems easy. It is often viewed as nothing more than an off-the-shelf purchasing activity. It seems so easy: just purchase the right module and be done with it.

In this column, I'll focus on switching regulator modules only. For reasons such as low efficiency and high heat dissipation, linear regulators nowadays are used primarily to supply specific subcircuits drawing only relatively low current.

POWER SUPPLY

For prototyping or some noncritical application, a switching regulator module alone might be the sole component of the power supply. While purchased modules provide well regulated output voltages, to ensure reliable system operation, there may be quite a bit of circuitry required to add to it. System specifications should characterize the available primary power in detail. Many bugs and inexplicable, intermittent failures have been caused by power supplies incapable of handling some exceptional conditions of the primary power.

Figure 1 is a schematic of a typical robust PS. Not all the components added to the switching regulator module will always be necessary. I'll refer to their purpose when discussing specific primary power conditions to be encountered.

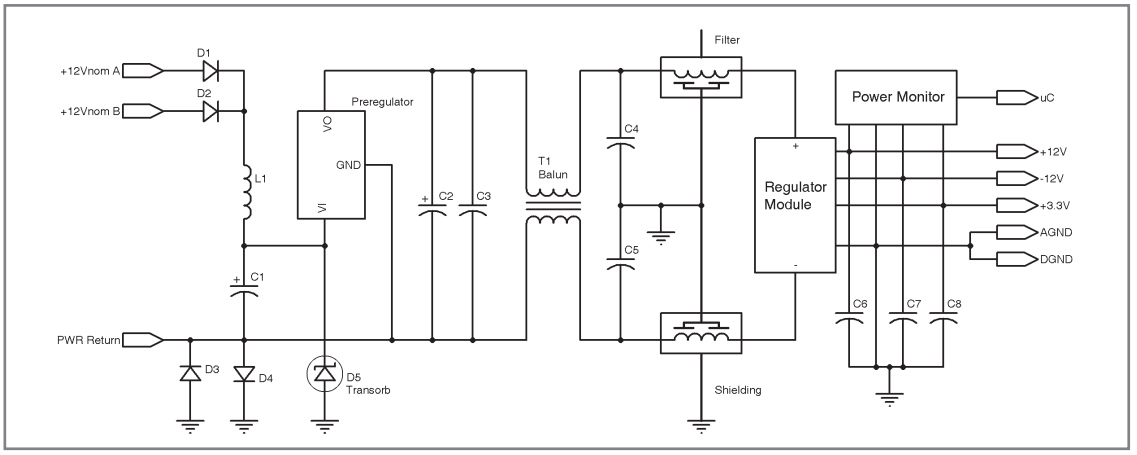


Figure 1—A full-featured power supply

Let's assume the power supply must deliver ±12 V for analog and 3.3 V for digital circuits of an automotive embedded controller subsystem. The primary power is the car's 12-V power distribution system, which is an alternator and a floating lead-acid battery. For a different type of the primary power, the design of the PS will vary depending on the power characteristics, but the process remains the same. To begin, you'll have to know the current draw of each generated voltage, its maximum allowed ripple and, together with the efficiency requirement, select an off-the-shelf switching regulator module. Sometimes you may have to use several modules to satisfy the specification, but the chance that you will have to design the regulator from scratch is fairly small these days.

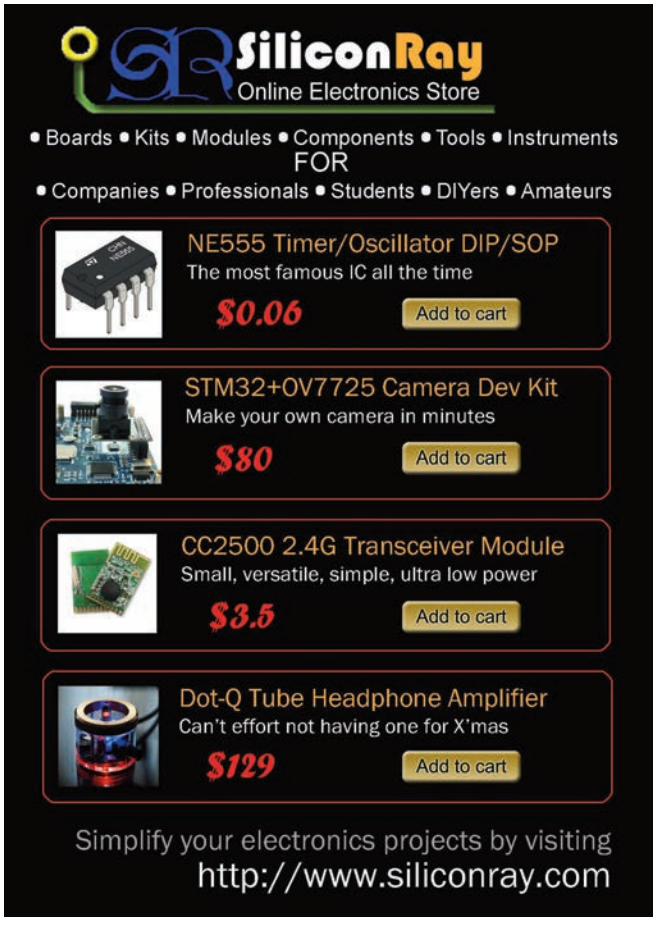
You will likely find several modules fitting the bill, some with their inputs and outputs isolated, others sharing the common (ground) terminal. Isolated modules are usually a bit more expensive, but if you can afford it, use them. I always prefer the isolated module, especially when sensitive analog interfaces are used. Isolation makes ground loop prevention much easier. (Refer to my column, "My Analog World: The Significance of Grounding," *Circuit Cellar* 244, 2010.)

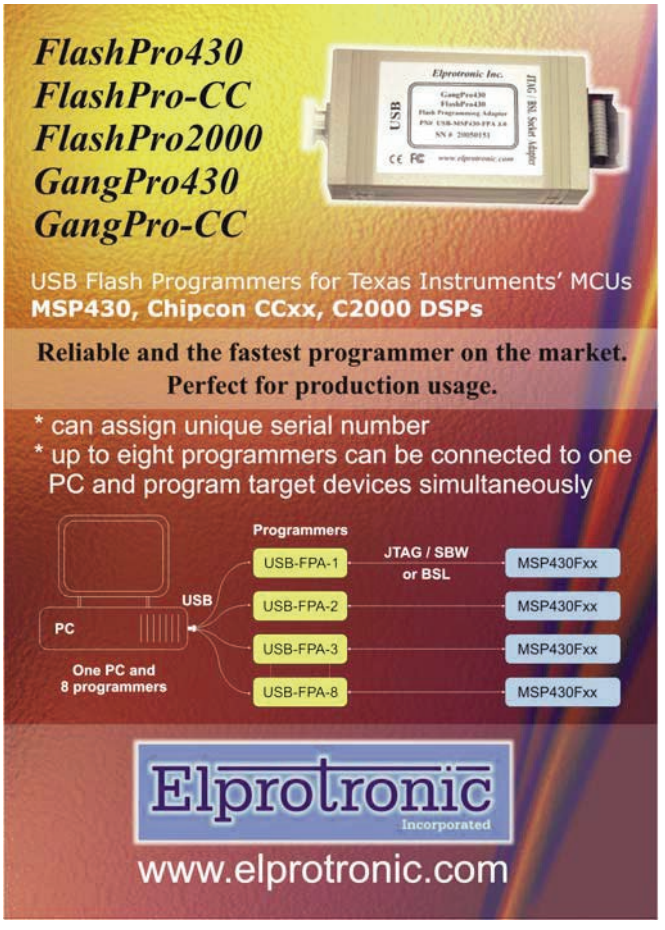
POWER DISSIPATION

Power dissipation is an important consideration. Some modules come with a built-in heatsink, others need an external one. Cooling electronics is crucial. In my opinion, there is no such thing as too much cooling. The hotter the electronics run, the worse their reliability will be.

When selecting the module, you will have to review its operating input voltage. For an automotive 12-V floating lead-acid battery system, the normal operating voltage will vary between 11 and 15.15 VDC. There will also be abnormal voltage swings, which may extend the primary power range from 10.25 to 16.2 V. Also, some subsystems require "emergency" operation, which drops the low voltage limit down to 9 V. Diodes D1 or D2 in Figure 1 provide isolation from the system power distribution. Without them, C1 would discharge into the primary power net and be essentially useless. Unfortunately, the diodes reduce the input voltage even further, typically by 0.6 V for regular rectifier diodes. Thus the regulator module must be rated to work with input voltage from about 8.4 V (with "emergency" operation) up to 15.6 V. Maximum ripple voltage on 12-V power distribution systems with a floating battery should not exceed 1 V_{PP}, but if it does, or the module cannot live with ripple, capacitor C1 and diodes D1 or D2 in Figure 1 will keep the ripple at an acceptable level. Photo 1 shows construction of the input part of the PS.

Momentary interruptions of the primary power can cause a lot of headaches. The normal requirement is that a 50-ms hiccup causes no upset to the subsystem. For this, C1





provides the necessary storage, while the isolation diodes D1 or D2 prevent C1 from being discharged into the power bus. Important subsystems must be impervious to power outages as long as 200 ms, and the safety-critical ones up to 1 s. This is when C1 may become so large as to be impractical. A common solution is to use two separate, independent, primary supplies and logically OR them through D1 and D2. It can be reasonably assumed that the voltage drop is unlikely to simultaneously occur in two independent primary power buses.

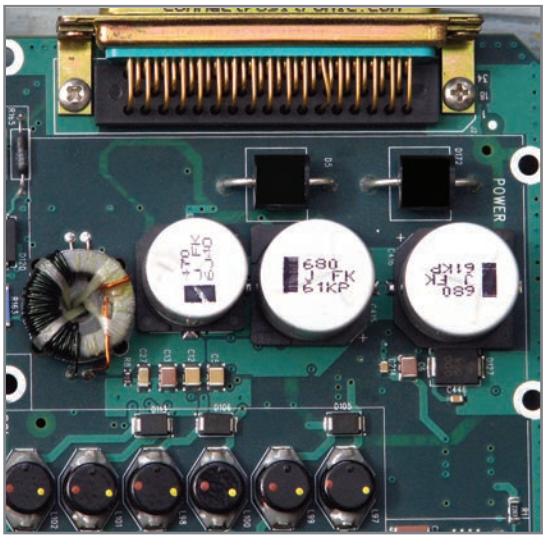


Photo 1—Power supply input. Isolation diodes D1 and D2, electrolytic capacitors forming C1, and the balun with its filter capacitors and chokes are clearly visible.

It is normal for the primary power bus to contain surges resulting from the normal system operation. They are the results of disturbances caused by load switching of other subsystems on the bus, regulators' remedial actions, etc. Their effect on the PS can be eliminated by C1 and L1. Figure 2 (not to scale) shows how to test PS immunity to normal surges. Changing between the voltage levels should take approximately one millisecond. Some subsystems may require immunity to increased surge voltages. Repeat the test several times while monitoring the PS performance. Unless permitted by the specification, no degradation of performance should be observed.

OPERATING CONDITIONS

Some subsystems may have to operate during cold engine start. In those cases, the battery voltage may go as low as 5 V for about 35 s. That means, considering the diode drop, a mere 4.4 V will be available to the regulator. That is a tough requirement. Power supply modules with such input voltage range are not inexpensive, but there are ways to meet the challenge.

To check the performance during cold engine start, start the PS at the nominal voltage, then drop to 5 V and increase by 0.3 V/s for 35 s. Then return to the nominal voltage.

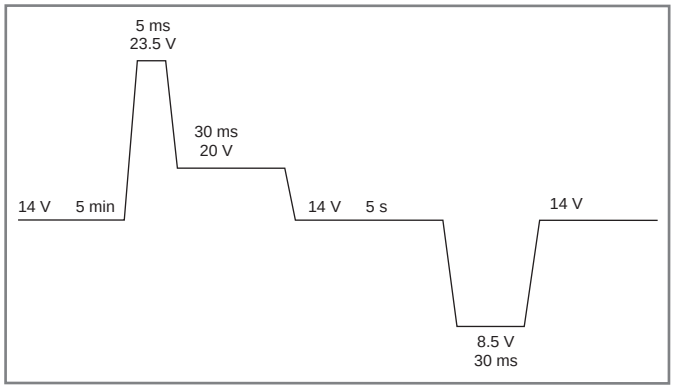


Figure 2—Normal surge test

During this time, monitor the subsystem performance. Some performance degradation may be permitted.

Another important condition to consider is low-voltage operation. What happens when the input to the regulator, for whatever reason, drops below its minimum operating voltage for, say, a few minutes? You need to study the module specification. Perhaps it shuts down automatically, or the input voltage needs to be monitored and the module externally shut down. If cold start is not required, the momentary undervoltage condition can be simulated by lowering the input voltage to 6 V for 7 s. After the return to the nominal operating voltage, the power supply must operate normally. A monitor, typically a Maxim Integrated Products MAX IC, as shown in Figure 1, ensures the subsystem is reset whenever the low input voltage condition occurs.

There are more aspects to the robust power supply design. We shall conclude the topic next month.

George Novacek (gnovacek@nexicom.net) is a professional engineer with a degree in Cybernetics and Closed-Loop Control. Now retired, he was most recently president of a multinational manufacturer for embedded control systems for aerospace applications. George wrote 26 feature articles for Circuit Cellar *between 1999 and 2004.*

RESOURCES

G. Novacek, "My Analog World: The Significance of Grounding," *Circuit Cellar* 244, 2010.

Department of Defense, Interface Standard, "MIL-STD-461 Requirements for the Control of Electromagnetic Interference Characteristics of Subsystems and Equipment," 1999, www.tscm.com/MIL-STD-461E.PDF.

Maxim Integrated Products, MAX IC, www.maxim-ic.com.

"MIL-STD-704, Aircraft Electric Power Characteristics," 2008, https://assist.daps.dla.mil/quicksearch/basic_profile.cfm?ident_number=35901.

Radio Technical Commission for Aeronautics (RTCA), DO 160, "New Revision G on Environmental Conditions and Test Procedures for Airborne Equipment," 2010, http://standardsforum.com/?p=464.



See where a single stroke of genius can change the course of tomorrow.

Calling all thinkers, creators, leaders, dreamers and innovators.
It's time to return to the home of unbridled jaw-dropping ideas.

2012 International CES®
THE GLOBAL STAGE FOR INNOVATION

PRODUCED BY CEA®

Tuesday, January 10–Friday, January 13, 2012
Las Vegas, Nevada | CESweb.org

REGISTER NOW

by Richard Wotiz (USA)



Digital X-Ray Sensors

Ever wonder how X-rays work? Here's a detailed look at imaging techniques, from their early inception to today's advances. You'll also learn how a CMOS imaging sensor functions.

This month, I'll look at various techniques used for X-ray imaging, from film to modern digital sensors. I'll also tear apart a CMOS imaging sensor and show how it works. It's a very different type of radiation sensor than the ionization chamber I discussed in my last column ("Ionization Detectors," *Circuit Cellar* 256, 2011). We'll even have a chance to look at some X-ray images and see just how useful they can be.

THE FIRST X-RAYS

X-rays have been used to peek inside solid objects for more than a century. It all started in 1895 when Wilhelm Roentgen accidentally discovered a beam of unknown rays emanating from a vacuum tube. He used the beam, now known as an X-ray, to take a photograph of his wife's hand (see Photo 1). Since that time, X-rays have been used to generate images on photographic film or photosensitized plates in medical, industrial, and other applications. Like other forms of ionizing radiation, X-rays, which are high-energy photons, are harmful to living tissue. Therefore, it's best to use the lowest possible beam intensity to produce a usable image. But film is not as sensitive to X-rays as it is to visible light. Making the film more sensitive would increase the graininess of the image, reducing its resolution. Instead, an X-ray film holder contains an image intensifier screen made of a special material that converts X-rays to light. With the screen, you need a significantly lower exposure to get the same image as with film alone.



Photo 1—This is the earliest known X-ray image, captured by Wilhelm Roentgen in 1895. It's called "Hand mit Ringen," or "Hand with Ring," and shows Roentgen's wife's hand wearing her wedding ring. (Source: National Aeronautics and Space Administration)

The first time digital imaging was used for X-rays was in the early 1970s, with the development of the first practical computerized axial tomography (CAT) scanner, now more commonly known as computed tomography (CT). It consists of an X-ray tube and an electronic detector at opposite ends of a rotating gantry, with a patient placed in the middle (see Figure 1). Data from the detector is stored as the gantry rotates. The platform under the patient moves slowly (either in discrete steps or continuously in a helical scan system). This forms a series of image slices, which are combined using various mathematical algorithms into a single 3-D representation of the patient's body. The detectors in CT scanners use the same type of scintillation material to convert X-rays into light as in the image

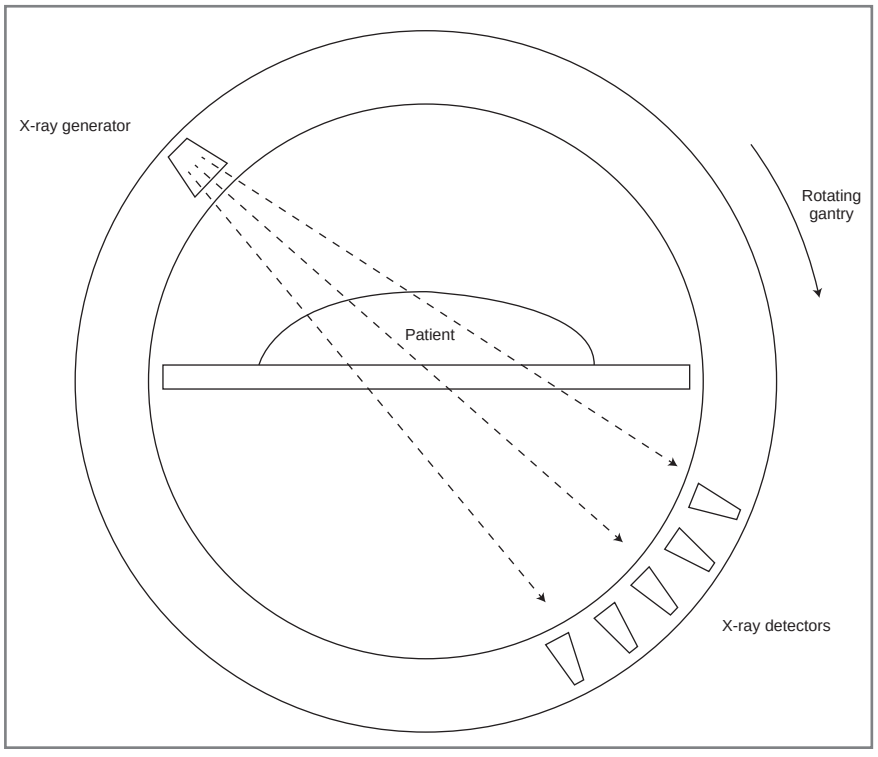


Figure 1—The basic layout of a CT scanner. The theory behind the mechanism is straightforward. The complexity is in the computations done to the data in order to create a 3-D image.

intensifier screen used for film. Early scanners used a photomultiplier tube to convert the light into electricity, while modern machines use photodiodes.

There is also an indirect digital imaging technique using photostimulable phosphor (PSP) plates. These are flat plates coated with a material that stores energy in areas that are exposed to X-rays. The latent image can be read out by scanning the plate with a laser beam, where exposed areas will emit light when hit by the laser. The scanner can capture the image and store it in a computer. It's almost like a laser printer in reverse. This technique is also known as computed radiography, since a computer is used to control the scanner, but it's not to be confused with computed tomography.

Dental X-rays use similar imaging techniques, but on a much smaller scale. When I go for a checkup, my dentist will take a set of intraoral X-rays using small pieces of film carefully placed in my mouth. The X-rays expose the film directly, without the image intensifier screen that's used in larger film holders. That results in a higher resolution image, since the thickness of the screen enables the light to spread out slightly, providing an image that's not as sharply focused.

DIGITAL SENSORS

Digital intraoral X-ray sensors were first developed in the 1980s. There have been many technological improvements since then, and many dentists now accept the images from modern sensors as a suitable replacement for film. Photo 2 shows the business end of a sensor. It's a bit larger than a piece of dental X-ray film, and it's also more rigid. The other end of the cable plugs into an interface module that connects to a computer's USB port.

Figure 2 shows the key elements of the system. The imaging sensor in this particular unit is a CMOS active pixel sensor (APS) array, though some other products use charge-coupled device (CCD) sensors instead. (Refer to the Resources section of this article for more information on some typical sensors.) CMOS sensors consume less power and can be mass produced more economically than CCDs, which has made them popular in digital cameras. Early development work on CMOS sensors in the 1990s came from research at the U.S. Jet Propulsion Laboratory. They wanted to create a compact, low-power imaging system for NASA for use in deep-space exploration, where minimizing power consumption is critical. At the same time, researchers in Scotland and Sweden were looking to produce a low-cost, single-chip imaging system. The results of these efforts, as well as the steady improvements in CMOS processing technology, gives us the sensors in use today.[1]

There is more pixel-to-pixel variation in the output signal of a CMOS imager than with CCDs, which produces more noise in the raw image data. Most of this is known as fixed-pattern noise that results from manufacturing process variations between the pixels and doesn't vary from one exposure to the next. The system compensates for this noise, as I'll explain later. Much of the image-capture logic is contained on the sensor chip, so it can acquire the X-ray image in real-time without any outside help. Since there's no connection between the sensor and the X-ray generator, it can't rely on an external trigger signal. There are extra photodiodes scattered in several locations around the sensor that aren't part of the main image array. These photodiodes are always active, and are used to sense when there is an incoming X-ray beam. This triggers the on-board capture

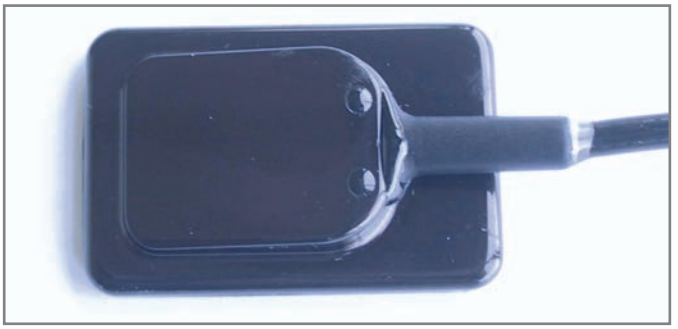


Photo 2—This is a bottom view of a digital intraoral X-ray sensor. It's well sealed to withstand the punishment it gets from being in a patient's mouth. Note the extra sealant that's visible where the cable enters the strain relief.

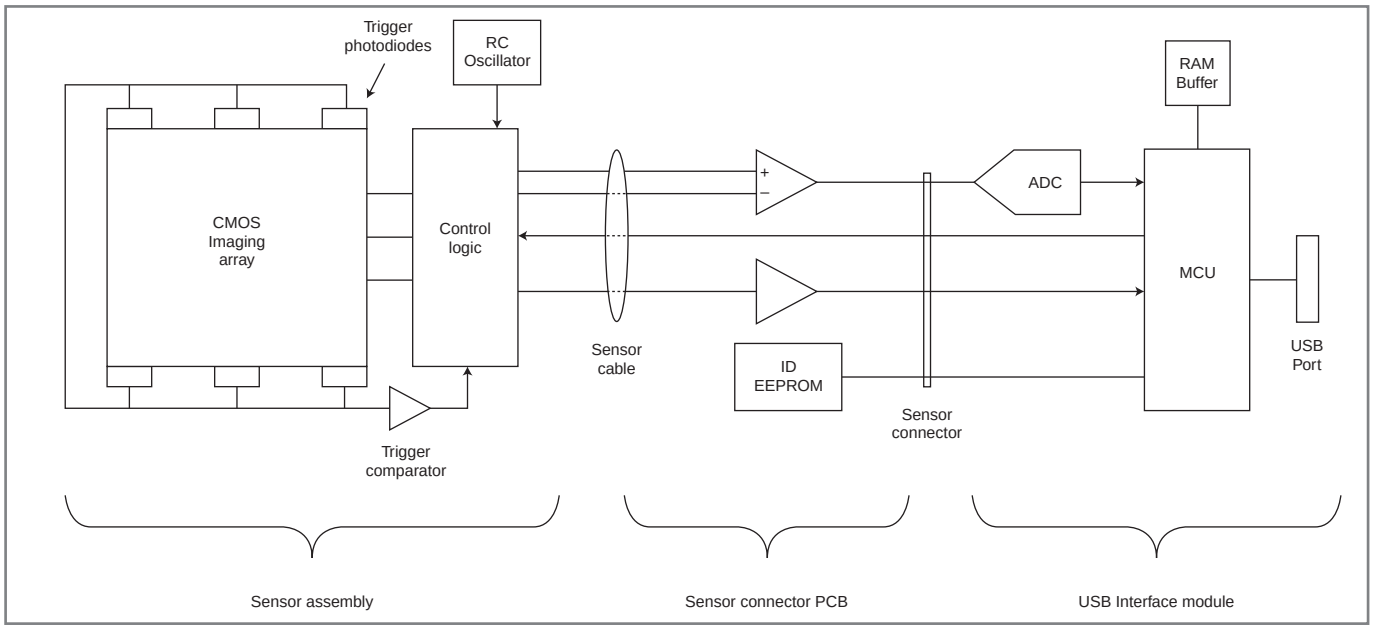


Figure 2—The sensor and USB interface. Everything shown on the sensor assembly, except for the RC oscillator, is contained within the CMOS imager chip. The chip outputs a differential analog signal, which is buffered on its way to a 12-bit ADC.

logic to acquire an image. Once the beam stops, the analog image data is sent the interface module where it's fed through a 12-bit ADC to a microcontroller and stored temporarily. Once this process is complete, the sensor acquires a second image, this time without an X-ray beam. The microcontroller subtracts this data from the first image to compensate for any fixed-pattern noise in the image sensor.[2] There can also be a gain error from one pixel to the next. Each assembled sensor module comes with a manufacturer-supplied calibration file that is used to adjust the gain of each pixel. The sensor module has a serial EEPROM containing a serial number that lets the computer identify the proper calibration file to use. Once the image data has been processed, it's stored on the host computer and displayed for the dentist to examine.

Figure 3 shows the construction of the sensor assembly. The top aluminum layer shields the electronics from outside noise, but is thin enough not to interfere with the X-rays. The CMOS sensor will directly respond to X-rays but, like film, it's much more sensitive to light. The scintillation layer converts X-rays to light so they can be picked up by the sensor. There are several different scintillation materials in use, including thallium-doped cesium iodide (CsI:Tl) and terbium-doped gadolinium oxysulfide (Gd_2O_2S:Tb), also known as Gadox. These are special materials that respond to incoming radiation by producing flashes of visible light

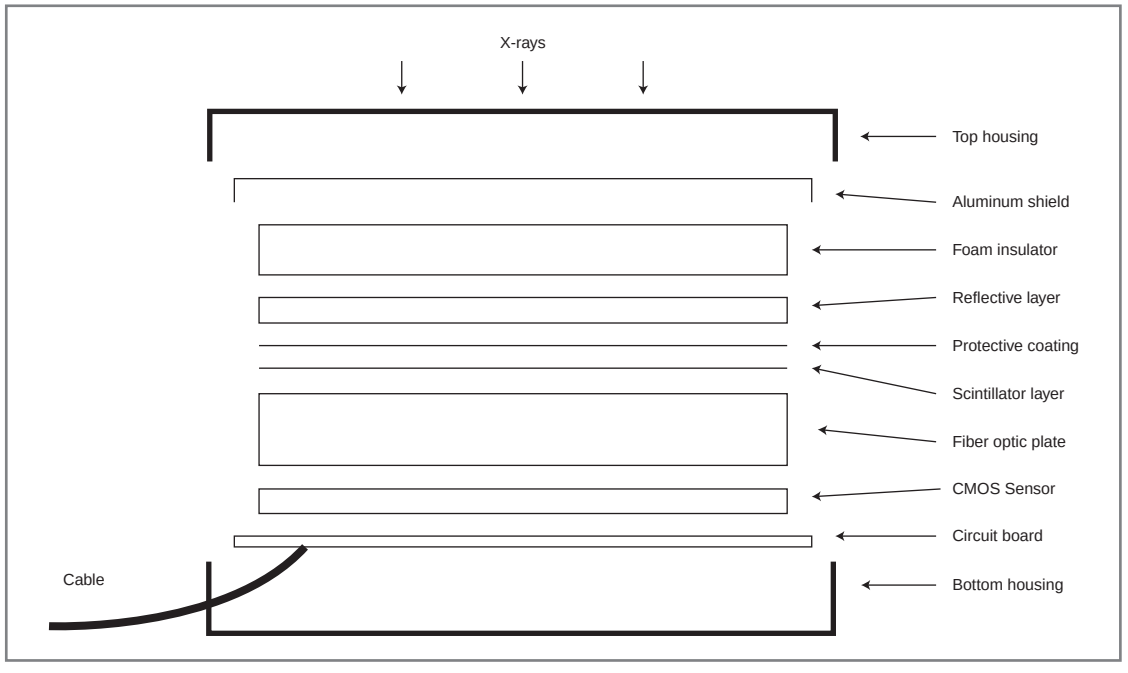


Figure 3—The construction of an X-ray sensor. Even with today's technology, it's still amazing how much can fit into such a tiny space.

APEC® 2012

February 5–9, 2012

Disney's Coronado Springs Resort
Orlando, FL

The Premier Global Event in Power Electronics™

Visit the Apec 2012 web site for the latest information!

www.apec-conf.org

SPONSORED BY

pels

psma power sources manufacturers association

IAS IEEE INDUSTRY APPLICATIONS SOCIETY



Photo 3a—Top view of the image sensor. I removed the aluminum shield, foam, and reflector so you can see the coating on the fiber-optic plate. The plate is glued to the sensor, so I couldn't disassemble it any further. **b**—The back side of the sensor board shows some of the smallest components I have ever seen. The ceramic board is more dimensionally stable than a fiberglass board would be.

whose brightness corresponds to the energy of the radiation. I wasn't able to find out which material this particular sensor uses, but CsI:Tl appears to be the most common among sensors from other manufacturers. It produces green light with an emission spectrum centered around 550 nm, and is one of the brightest scintillation materials available. It's also slightly hygroscopic, so it needs to be sealed with a moisture-resistant layer.

Once the scintillation layer produces a flash of light, it's important to direct it toward the CMOS imager without enabling it to spread out, or the image would become blurry. The fiber-optic plate handles this with a series of vertically oriented microscopic light fibers that are bundled together. It's glued to the sensor's front surface and guides light directly to the sensor. The plate also absorbs any X-rays that aren't trapped by the scintillation layer. This keeps them from reaching the CMOS sensor, preventing them from creating additional noise in the image data. X-rays will also degrade the sensor over time by increasing the sensor dark current and shifting the transistor threshold voltages.[3]

LOOKING INSIDE

Let's examine the sensor in more detail. They're a bit too expensive to sacrifice a good one, but I was able to obtain a failed sensor, thanks to Dr. Catherine Brennan, DDS. Photo 3 shows both sides of the sensor assembly. The fiber-optic plate is glued firmly to the sensor. I wasn't able to remove it, so you can't see the front surface of the CMOS imager. Photo 3b shows the back of the circuit board. It's made of ceramic, which is more solid and less affected by temperature changes than a more common fiberglass-based material. Note that there's very little circuitry needed for the sensor. The two tiny ICs are an RC oscillator and a logic-gate buffer. You can

AP CIRCUITS
PCB Fabrication Since 1984

As low as...
$9.95 each!

Two Boards
Two Layers
Two Masks
One Legend

Unmasked boards ship next day!

www.apcircuits.com

VISA MasterCard PayPal IPC MEMBER ASSOCIATION CONNECTING ELECTRONICS INDUSTRIES BBB

System on Module
Internet Appliance Engine

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

2.6 KERNEL
Microsoft Windows CE

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

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

Since 1985 OVER 24 YEARS OF SINGLE BOARD SOLUTIONS

EMAC, inc.
EQUIPMENT MONITOR AND CONTROL

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

see several small dark rectangles near the wire connections, which are resistors directly screen printed on the ceramic board with resistive ink. Screened resistors aren't very accurate unless they're trimmed, which these don't appear to be. Here they are used as pull-ups and series ESD protection resistors, where their exact values aren't critical. Most of the other passive components are 0402 size. I can't imagine trying to debug a product with parts that small! The far end of the sensor's cable is attached to a small printed circuit board (PCB) with a row of edge fingers. It just holds some analog and digital buffers, as well as the serial number EEPROM, and plugs into a connector on the USB interface module.

This particular sensor has a 30 mm × 20 mm active area and a 40 µm pixel pitch, giving a resolution of 750 × 500 pixels. Other sensors have pixel pitches as small as 19 µm, but, as with digital cameras, there are so many factors that go into producing an image that it's hard to judge quality simply from the numbers. You can find other dental

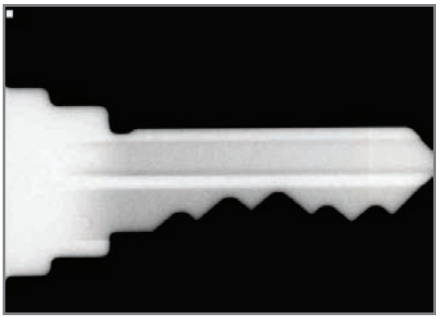


Photo 4—Here you can finally see some results. The brass key only partially blocks X-rays, which is why you need to use a denser material, such as lead, to completely block them.

sensors with active areas from 24 mm × 18 mm up to 36 mm × 26 mm.

The strain relief on the end of the sensor wire is the most frequent point of failure. That's not surprising, especially for the smaller sensors that are used primarily in children's mouths. Some of the newer ones have replaceable cables, and there are even a handful of wireless sensors available that eliminate the problem entirely. But then you need to worry about monitoring signal quality and replacing batteries.

TAKING PICTURES

Photo 4 shows an image taken with a functioning sensor. You can see that enough of the X-ray beam penetrates the 0.1″ thick brass key to show some of the groove detail. If you look closely, you can see a faint vertical stripe near the tip that's slightly lighter than the rest of the key. That's a variation in the metal density from when the key blank was formed. It makes it clear how X-rays can be useful to check metal parts for quality or uniformity.

Photo 5 shows the experimental setup. The X-ray generator was set for 8 mA and 60 kV peak (kV_p), with an exposure time of 80 ms. The tube current sets the intensity of the beam, which corresponds to the brightness of a visual image. The generator voltage determines the maximum energy of the X-ray photons in electron volts (eV). This corresponds to their wavelength, or the color in visual terms. The

Get PUBLISHED. Get NOTICED. Get PAID.

Circuit Cellar feature articles are contributed by professional engineers, academics, and students from around the globe. Each month, the editorial staff reviews dozens of article proposals and submissions. Only the best make it into the pages of this internationally respected magazine.

Do you have what it takes?

Contact **C. J. Abate, Editor-in-Chief,** today to discuss the embedded design projects and programming applications you've been working on and your article could be featured in an upcoming issue of *Circuit Cellar* magazine.

editor@circuitcellar.com

CIRCUIT CELLAR

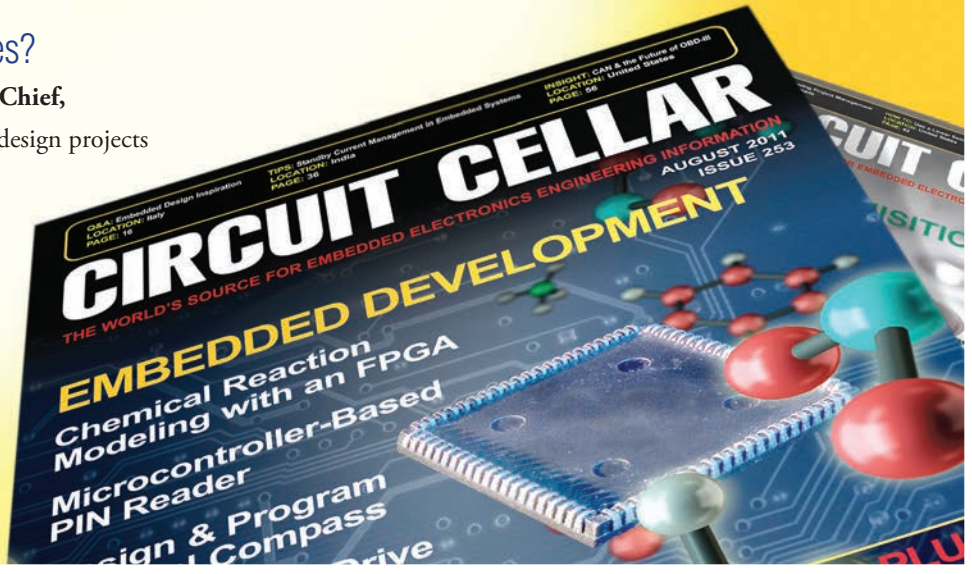




Photo 5—The X-ray machine is ready to take the image in Photo 4. That's a sample from my growing Orphan Key collection.

wavelength λ is given by λ = 1.24/E, with λ in nanometers and E in kilo-electron volts (keV). That gives us a value of 0.02 nm for 60 keV. A thin layer of aluminum at the output of the generator blocks any lower energy "soft" X-rays that are produced, since they would be absorbed by the skin and wouldn't contribute to the captured image.

Typical sensor exposures are about one third to two thirds of film exposure times. The shorter times aren't a problem for modern X-ray machines, but older models that were designed for film can't always be set so short. That's not surprising, since the contact bounce time on the relays used in older machines would likely be a significant fraction of the exposure time. The system used here can be set from 10 ms to 3.2 s, with a tolerance of 5% + 1 ms.[4] The exposure time translates directly into radiation dose, so it's better to use the shortest time possible that produces an acceptable image. That's called as low as reasonably achievable, or ALARA. It's a principle that applies to radiation exposure with a goal of using the minimum dose that will reasonably provide the necessary diagnostic information. In general, the higher the dose, the greater the level of detail in the image. That means there's a tradeoff between how much information is really necessary, and how much risk that information is worth. In our case, the decrease in dose when going from film to a digital sensor is also accompanied by some loss of detail, since even the highest resolution sensor can't match the resolution of film. I'll have to trust my dentist to know whether or not that difference is significant when it comes to looking inside my teeth.

Now that you have a better understanding of what goes on behind the scenes in your dentist's office, perhaps the next visit will be less scary!

Richard Wotiz has been taking products apart ever since he was old enough to pick up a soldering iron. He's been helping others put them back together since 1991, when he started his design consulting business. Richard specializes in hardware and software for consumer products and children's toys. He can be reached at rw601@spiraltap.com.

REFERENCES

[1] E. Fossum, "CMOS Image Sensors: Electronic Camera-On-A-Chip," IEEE Transactions on Electron Devices, 1997.

[2] J. Haystead, "CMOS Sensors Help Dental Imaging Systems Outperform X-Ray Film," *Vision Systems Design*, 1998, www.vision-systems.com/articles/print/volume-3/issue-12/features/feature-article/cmos-sensors-help-dental-imaging-systems-outperform-x-ray-film.html.

[3] Rad-icon Imaging Corp., "AN06: Detector Lifetime and Radiation Damage," 2002.

[4] Planmeca Oy, "Planmeca Intra X-Ray Unit User's Manual," 2005.

RESOURCES

Hamamatsu Corp., Dental Applications, http://sales.hamamatsu.com/en/products/solid-state-division/image-sensors/dental-ccd.php.

Saint-Gobain Crystals, Scintillation Products, Thallium Activated Cesium Iodide, www.detectors.saint-gobain.com/CsI(Tl).aspx.

R. Wotiz, "Ionization Detectors," *Circuit Cellar* 256, 2011.

Designed a Brand New Device?

Learn to search online databases to see if your device is cutting edge and may be eligible for a patent by using this new instructional DVD!



MIDWEST PATENT SERVICES

info@midwestpatentservices.com

www.midwestpatentservices.com

Elektor Print

Classical reading: on paper



+

Elektor Digital

Contemporary & innovative reading: on PC, notebook or tablet

=

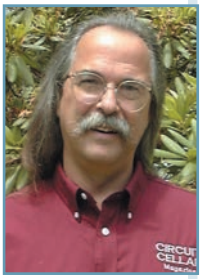
Elektor PLUS

Comprehensive reading: at home and on the road

Read Elektor with the premium PLUS subscription!

Subscribe now or upgrade: www.elektor.com/usa

by Jeff Bachiochi (USA)



Quantifying Precipitation

Construct a Device to Measure Rainfall Volume

This tip-bucket mechanism uses a Hall-effect sensor and a microcontroller with debugging capability, LEDs, and configuration jumpers to measure rainfall volume. The sensor's open-collector output is connected to the microcontroller's interrupt input.

One of the longer-term projects I'm working on for my yard is a weather station. The WeatherBug gadget on my Windows sidebar provides a fairly complete look at the present conditions. But the closest reporting location can be a good distance away from the physical location, and a few miles can often make a big difference in your exposure to Mother Nature's wrath.

As a child, I can remember going outside to play in the rain. My side of the street was being flooded from a cloud-busting downpour while just across the street, the parched lawns were screaming for a bit of moisture. Our weather patterns control water—the most important commodity on earth—and it is served up in a diverse assortment of pleasing and distasteful recipes.

Voluminous winter snows test roofing capacities and create impassable roadways. Snow shoveling causes back pain and can lead to heart attacks for those not up to the task. Hail stones smash and dent automobiles and homes. An abundance of rain can swell rivers into flooding torrents that pose a threat to lowland areas protected by dams and levees. While the weather can be too generous with its assets, it can also withhold them to the point of equal calamity. A lack of moisture means certain death to any plant or animal that can't sustain prolonged periods of dehydration.

It seems like anyone you talk to can tell a story about either too much or too little water. You rarely see a public outcry with having to pay as much for water (a buck a liter) as for gasoline. Water, as a commodity, will surely be one of the best investments for the future.

TIPPING THE SCALES

When it comes to measuring rainfall, the National Oceanic and Atmospheric Administration's (NOAA) national weather service reports for the year 2010 that the continental U.S. had total of rainfall amounts from less than 1″ to more than 80″. The arid southwestern region

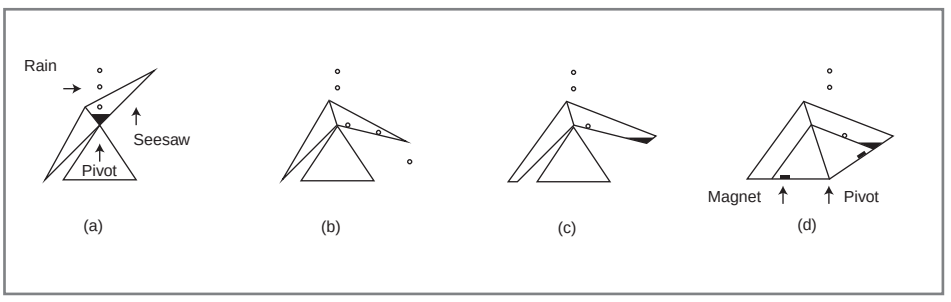


Figure 1—This is the progression my paper test models went through before I settled on a working design.

(California and Nevada) claims the least rainfall, while the northwestern Pacific costal area (Washington, Oregon, and California) received huge amounts of precipitation. The midwest region had less than 40", while the south and east coasts averaged more than 40".

Precipitation can be measured in many ways, including standard and automatic gages, radar and satellite feedback, disdrometer, and laser sensors. By far, the simplest is the standard gage made up of a vertical-walled container (preferably a clear material) with a depth scale mounted on the side. The disadvantage of this gage is that it must periodically be physically emptied to remain relevant. The automatic—or tip bucket—gauge is self emptying, but it can only measure in gulps. The bucket's size determines the amount of water it takes to create an unbalance enabling the bucket to tip and empty. This apparatus must also contain some method of keeping track of the number of tips the bucket has completed.

I experimented with a number of designs for a tip-bucket mechanism. I like modeling out of cardboard because you can get some quick results with just a few household items such as cereal boxes and some cellophane tape. In Figure 1, you can see the progression of the design. While cardboard isn't exactly waterproof, the model will last long enough to see the strengths and weaknesses of the design. In Figure 1a the simple seesaw (trough with central baffle) on top of a pivot point shows how water droplets can congregate. The resulting pool of water may never become unbalanced enough to tilt down, dump its contents, and enable the other side to fill.

By bending the seesaw, as shown in Figure 1b, the water droplets are forced away from the center. The farther away from center they roll, the more effect they have on the seesaw's balance. If the seesaw was long enough, one drop could tip the scale. This would also depend on the weight of the seesaw. The better approach is to collect the drops at a fixed distance. In Figure 1c, the ends of the seesaw are modified to create pooling areas.

Finally, with an eye toward fabrication, the bucket was simplified, which also moved the pivot point (see Figure 1d). This turned the analog seesaw into a digital one, that is, one that only has two stable positions (no potential balancing). With a working design, I used my PC to draw a 2-D representation. Then I could accurately cut out and fold a design from a printed template. This enabled me to experiment with different materials. While cardboard can be waterproofed with paint, I wanted to try using some plastics.



Photo 1—The container funnels collected rain into a diamond-shaped seesaw. When a sufficient amount of water upsets the balance, the seesaw tips, dumping the collection bucket and exposing the opposite bucket for future collection. A Hall-effect switch under the container senses the seesaw's position.

BONDING WITH PLASTIC

It just so happened that my wife, Beverly, had recently brought home Chinese food and it came in clear plastic containers. These are a very lightweight polystyrene (PS <6>), the same material that "U-build 'em" plastic model kits are made from. I still have a collection of modeling tools, so I had the glue needed to bond this stuff together. While this is tough stuff, it does not like to be bent too sharply. At a 90° angle it will snaps along the crease. I cut out the bucket template using household scissors and made folds using a small radius. The plastic can be easily bent and the model glue clouds the plastic slightly, but the bucket weighs hardly anything, so it should not take too much water to counteract the bucket's weight.

I was on a roll now, so I decided to make the outer enclosure out of see-through material too. The packaging material is a bit flimsy for an enclosure. Since strength is important for this project, I used a CD case (also polystyrene). The design of this enclosure utilizes the edges of the CD case (right-angle edges) to create strong corners. The area on the top of the enclosure comprises the total collection area. Any water that falls upon it will wash down its slanted sides, which form the entry funnel to the bucket located inside (see Photo 1). By definition, the container will be exposed to the weather, so the electronics and associated bucket sensor must be weatherproof. One of the best sensors to use when a device is exposed to the weather is a magnet and a Hall-effect sensor. Neither requires exposure to the elements nor direct contact. Many Hall-effect sensors are sensitive enough to detect the magnetic field of a small magnet over a small gap. The presence of a magnetic field will change the logic level of the Hall-effect sensor.

COUNTING TIPS

If you've ever waited or bussed tables, you know that at the end of the night everyone pools their tips and you go home with a few extra dollars to compensate for the low hourly wage. Some nights the total tips are significant. To determine the amount of rain falling on the gauge we need to count every tip (of the bucket). By sealing two magnets into opposite sides of the bucket (as shown in Figure 1d) they will counter balance one another and enable a single Hall-effect sensor to determine when a tip occurs. The sensor's open-collector output is connected to an interrupt input of a microcontroller that will do the counting. I want to design a standalone gage that becomes a slave SPI device to whatever will make use of the data. A six-conductor

interface contains power and ground along with the four SPI control and data lines, SCK, SDI, SDO, and SS. An eight-pin microcontroller could suffice for this job, but I used a 14-pin device so I could have debugging capability as well as a couple of LEDs and configuration jumpers (see Figure 2). The entire circuit (including the sensor) can be sealed into a tiny and inexpensive Hammond enclosure.

Having a few extra I/Os can sometimes save the day, so using a Microchip Technology PIC16F1823 microcontroller (less than $1 in hundreds) was a good compromise between cost and size for this project. PORTA can have any of its inputs configured as an interrupt source to the microcontroller on a rising or falling edge. Each time the bucket tips from one side to the other, the magnetic field of one magnet will either enter or leave the sensitive area of the Hall-effect device, creating a low or a high output. This will interrupt the microcontroller enabling a predetermined volume of water to be added to the totalizing register. Let's see what's needed here.

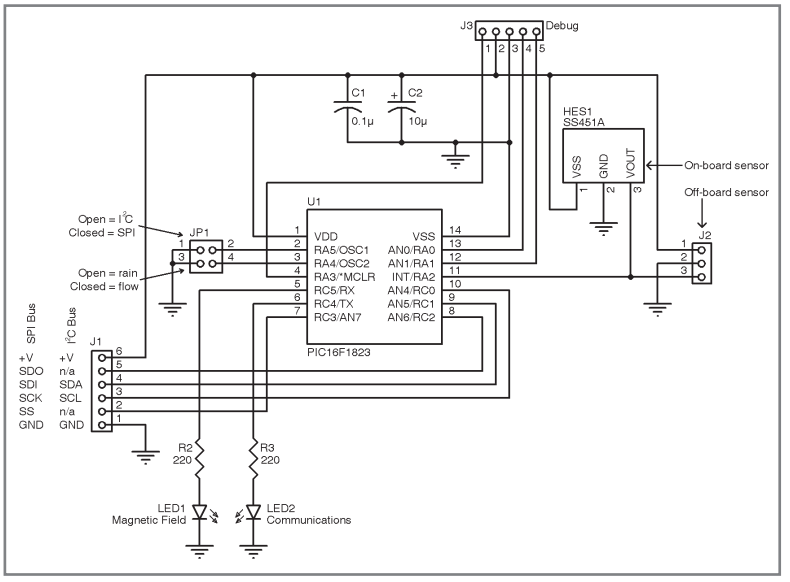


Figure 2—Using a low-pin-count (LPC) microcontroller creates a smart sensor capable of reporting total calculated rainfall of water flow. The circuit uses a Hall-effect sensor to count gulps of liquid swallowed by the presence of a magnetic field.

All you need is a ruler if you want to know the rainfall in inches and want to measure how much is collected in a vertical-sided container. You just measure the height of the collected water, the diameter and even the shape of the container are not important. However, if you want to calculate the rainfall in inches using volume, it is essential to know the area of collector's opening, since there is a direct relationship between area and volume. If the collection vessel is a 1″ square tube and has collected 1″ of water, then its Area = Length × Width or A = 1 × 1 and Volume = Area × Height or V = A × so Height = 1/Area (or 1:1 in this case). Therefore, we need to know the area of the collection vessel to be able to calculate a height in inches. Its rectangular shape makes this calculation easy (in this case, 4.65 × 1.10 = 5.115 square inches). The collection area is greater than five times that of our 1 cubic inch example. So to find the height in inches for this volume, we need to divide it by 5.115 (Height = 1/Area).

The tip bucket will measure the rainfall volume by indicating each time it takes a small gulp. To calculate this volume, we need to know what a gulp is. There are many factors that must be taken into account to calculate a gulp: weight of water, weight of the seesaw, length of one see (or is that saw?), and friction, etc. It is much easier to just measure it by pouring in a known quantity and counting the gulps. One liquid ounce is approximately 1.8 cubic inches. I filled the kitchen measuring cup with 10 oz of water, slowly poured the water into the top of the collector, and the bucket started filling and dumping left (tip 1), right (tip 2), left again (tip 3), on up to a total of 12 tips for 10 oz.

If 1 oz = 1.8 cubic inches, then 10 oz = 18 cubic inches. I had 12 tips for 18 cubic inches. That's 0.67 cubic inches per tip. Now remember, we need to divide this by 5.115 to get the relationship between 1 square inch and the total area of the collector. So one tip = 0.67″/5.115 or 0.13″). Every tip triggers an interrupt to the microcontroller and the interrupt code is responsible for incrementing the total rainfall by 0.13″.

Register	Access
0—Total volume (0–255 inches)	R/(W)
1—Total volume (0–99 hundredths of an inch)	R/(W)
2—Volume per tip (0–255 inches)	R/W
3—Volume per tip (0–99 hundredths of an inch)	R/W
4—Device (ID number)	R
5—Version (major hexadecimal: minor hexadecimal)	R

Table 1—In this registered implementation, an indirect pointer is updated as its part of the SPI command byte (see bits 5 and 6). The indirect pointer enables access to the available registers. In SPI mode, the total volume is not directly writable. Neither modes enable Device and Version data to be modified.

INTERFACES

SPI and I²C are popular interfaces. Their cultish following is evenly split amongst engineers. They both use the same peripheral and thus their associated interfaces often occur on the same I/O pins of any supporting microcontroller. In this project, both are supported to enable the module to be integrated into almost any situation. The SPI supports Mode 0 using SCK, SDI, SDO, and SS, while the I²C requires only SLK and SDA. Configuration Jumper 1 determines which interface will be used. The registers associated with communications are listed in Table 1.

In SPI mode, the first byte sent is always the command byte. The three most-significant bits (MSBs) of the command determine its function (see Table 2). A SPI write will therefore consist of sending the command byte and two data bytes. The SPI receiver will always contain a dummy value of 0x55 for each byte sent during a write command.

A SPI read will consist of sending the command byte and three dummy data bytes. The SPI receiver will always contain a dummy value of 0x55 for the first two bytes of a read command followed by two bytes of actual data. TotalVolume and VolumePerTip are both multibyte registers that contain an integer value and a fractional value of the data. Only legal commands will be acted upon, so while you could attempt to write to a read-only register, no action will be taken.

Command	Bit 7 1=READ 0=WRITE	Bit 6	Bit 5
Device:Version	1	0	0
VPTI:VPTF	0/1	0	1
TVI:TVF	1	1	0
TVI:TVF (then clear)	1	1	1

Table 2—The first byte sent in a SPI communication is the command byte. It determines the function to be performed. Bits 6:5 will set an indirect pointer for accessing data. The Volume registers consist of two 8-bit values. The MSB is the integer portion (in inches) and the LSB is the fractional portion (in hundredths).

In I²C mode, the device is also treated as a registered device using an indirect pointer to access to data. The first data value of any I²C write will update the indirect pointer. Additional data (within the same write command) will be loaded into the register pointed to by the indirect register. After each data access, this pointer will automatically increment (and wrap around if necessary). I²C read commands use the pointer in the same way and will continue incrementing such that additional data will come from successive registers. Most reads should begin with a write to set the pointer, just to be safe.

For this project, I am using a Honeywell International SS451A Omnipolar Hall-effect digital position sensor in a TO-92 style package. I've placed a sensor on the printed circuit board (PCB) but a connector also enables the device to be mounted externally. Jumper 1 selects between SPI and I²C communication, while Jumper 2 selects between the rain and flow gage. The constants are preprogrammed into EEPROM. After reset (power up) and once a value has been read from EEPROM into its respective RAM register, should the user write a new value to this register, it will also be saved in the nonvolatile EEPROM. In this way, the constant used to calculate a total can be permanently changed if necessary. LED1 indicates when a magnetic field is sensed and LED2 indicates communication activity.

WATER OVER THE DAM

Some of you may remember me talking about my water system in the past. Eight neighborhood families all draw water from the same well that happens to be on my property. Guess who gets the call when the water stops flowing? I thought it might help to keep track of the water usage, and searched the Internet for some gage that could be put on the 2" PVC water main. Most of what's available starts at approximately $500. Sounds like I need to make a trip to the hardware store.

PVC water pipe is easy to cut and glue and comes in many sizes. I also found some squarish stock in the rain gutter section that will provide some flat material. From the flat areas I cut



Photo 2—Tiny magnets on the tips of each paddle are sensed by a Hall-effect sensor that is sealed into the wheel's support tube. Water passing through the supply tube spins the paddle wheel. The three tubes are secured inside a larger 4" "T" that correctly orients the tubes and creates a proper water seal.

HUMANDATA

FPGA/CPLD Stamp Module
PLCC68 Series
Easy and Quickly Mountable Module

FPGA Module PLCC socket mountable
• 50 I/Os (External clock inputs are available)
• 3.3V single power supply operation (Voltage converters for auxiliary power supply are built-in)
• Separated supply-inputs: power, I/O drivers
• JTAG signal
• All PLCC68 series have common pin assignment
• Very small size (25.3 x 25.3 [mm])
• RoHS compliance
• MADE IN JAPAN

XILINX PLCC68 Series

XP68-01 Spartan-6 PLCC68 FPGA Module
NEW Spartan-6 PLCC68
XC6SLX16-2CSG225C
16Mbit Configuration SPI-ROM
Two User LEDs
One User Switch(Slide)
RoHS compliant

XP68-02 Spartan-3AN PLCC68 FPGA Module
NEW Spartan-3AN PLCC68
XC3S200AN-4FTG256C
FPGA internal configuration ROM
Two User LEDs
RoHS compliant

ALTERA PLCC68 Series

AP68-01 MAX II PLCC68 CPLD Module
NEW MAX II PLCC68
EPM240F100C5N
EPM570F100C5N
External Clock inputs
RoHS compliant

AP68-02 MAX V PLCC68 CPLD Module
NEW MAX V PLCC68
5M570ZF256C5N
External Clock inputs
On-board Voltage regulator
RoHS compliant

AP68-03 Cyclone III PLCC68 FPGA Module
NEW Cyclone III PLCC68
EP3C5U256C8N
EP3C10U256C8N
4Mbit Configuration ROM
Two User LEDs
One User Switch(Slide)
RoHS compliant

Over 100 varieties of FPGA/CPLD boards are available.

See all our products at:
www.hdl.co.jp/CC/

HuMANDATA LTD.
E-mail: s2@hdl.co.jp
Fax: 81-72-620-2003

out eight small 1-3/4″ paddles. Each paddle has a flat end that I shaped to fit and glue around a small brass tube. The finished paddle wheel fits into a large notch cut out of the end of a length of 2″ pipe and spins freely (see Photo 2). This will be suspended within a much larger 4″ "T" fitting with 4-to-2″ reducers. A baffle inside the "T" restricts water flow so that it is directed mainly toward the paddle wheel. A small magnet is epoxied to the outside tip of each paddle. I found some great, tiny, rare earth magnets at the local hobby shop. A single Hall-effect sensor is located on the dry side of a plug in the 2″ pipe supporting the paddle wheel. The magnetic field passes through this plug but no water can pass through.

The finished flow sensor looks a bit clunky but should withstand the pressure of the water system. My original thought was to mount the 2″ pipe supporting the paddle wheel and sensor using a compression fitting so it could be removed if necessary. I was not able to find the right fittings to make this work and withstand the system pressure. So, with a heavy heart, I glued it all permanently shut. While there is little inside that can fail, it cannot be opened for repair.

Calibrating this flow gage was easier than the rain gage. With a garden hose coupled to the inlet, I could run water through the flow gage and count the output transitions as each paddle passed the Hall-effect sensor. After a five-gallon pail was filled, I found that I had a total of 500 counts (500 divided by 5 = 100 counts per gallon, or approximately one hundredth of a gallon). This constant is added to the program for use when the circuitry is measuring gallons of flow as opposed to inches of rain.

HALL-EFFECT VS. REED SWITCH

For years, reed switches have been used to detect the presence of a magnetic field. They are small and fragile but inexpensive and tend to fail from fatigue and residual magnetism. Tiny switching contacts within their glass encapsulation shells can handle a variety of signals including analog and audio.

The solid-state Hall-effect sensors have no moving parts to wear out, but do require about 5 mA at 3 to 24 V. Most provide an open collector output that switches state when in the presence of a magnetic field. These are available in not only various sensitivities but also different magnetic characteristics. Omnipolar varieties react to the presence of any magnetic field (north or south) flowing through the sensor area. Unipolar sensors react only to a specific magnetic field (north or south) flowing through the sensor area. Bipolar latching sensors remain in their last state until the opposite field flows through the sensor area (alternating north and south). You can see why the Hall-effect device has become so popular as a non-contact (magnetic) proximity sensor.

It's amazing what can be accomplished with a single (bit) sensor. Unfortunately, there is often a whole lot more work in supporting a function then in the function itself. You shouldn't be frightened off from implementing an idea just because you can't find an off-the-shelf solution. I'd love to have access to a real shop full of tools. However, I find I can get by with a few basic tools from the cellar workbench and the use of the dining room table for assembly (while my wife is out shopping).

I found an old serial-to-I²C dongle in my box of gadgets to test this device. If you don't have something like this, experimenting with this project might be useless. So, next month, I'll present a useful project to give you access to a slew of interfaces rolled into one. Then there will be no reason to put off that project you've been thinking about but had no way of communicating with.

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

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2012/258.

SOURCES

SS451A Omnipolar Hall-effect digital position sensor
Honeywell International, Inc. | www.honeywell.com

PIC16F1823 Microcontroller
Microchip Technology, Inc. | www.microchip.com

NEED-TO-KNOW INFO

Knowledge is power. In the computer applications industry, informed engineers and programmers don't just survive, they *thrive* and *excel*. For more need-to-know information about some of the topics covered in this article, the *Circuit Cellar* editorial staff recommends the following content:

—

Multifunctional Wireless Alarm
by Carl Smith
***Circuit Cellar* 194, 2006**

Carl's multifunctional wireless alarm system can monitor everything from the doors to the sump pump in your house. The system features an MC13192 SARD board and several wireless sensors. When the alarm is activated, simply place a call to your house to obtain a status report. Topics: Alarm, Wireless, MC13192 SARD, MC9S08GT, Water Level, Temperature, DTMF, Emic, CH1837A, SSI204, MPX2010GS, Accelerometer, MMA1260D

—

Go to *Circuit Cellar*'s webshop to find this article and more: www.cc-webshop.com

A GOLDEN OPPORTUNITY

The price of gold just went *down* at CIRCUIT CELLAR!

Purchase Circuit Cellar's USB archive, **CC Gold** at $25.00 off the regular price and load it up with select add-ons for only $5.00 each.

Available add-ons:
Previous Contest Projects
Elektor Back Issues
audioXpress Back Issues

Archive includes:
All Circuit Cellar issues in pdf format that are in print through date of purchase. Also includes associated project code.

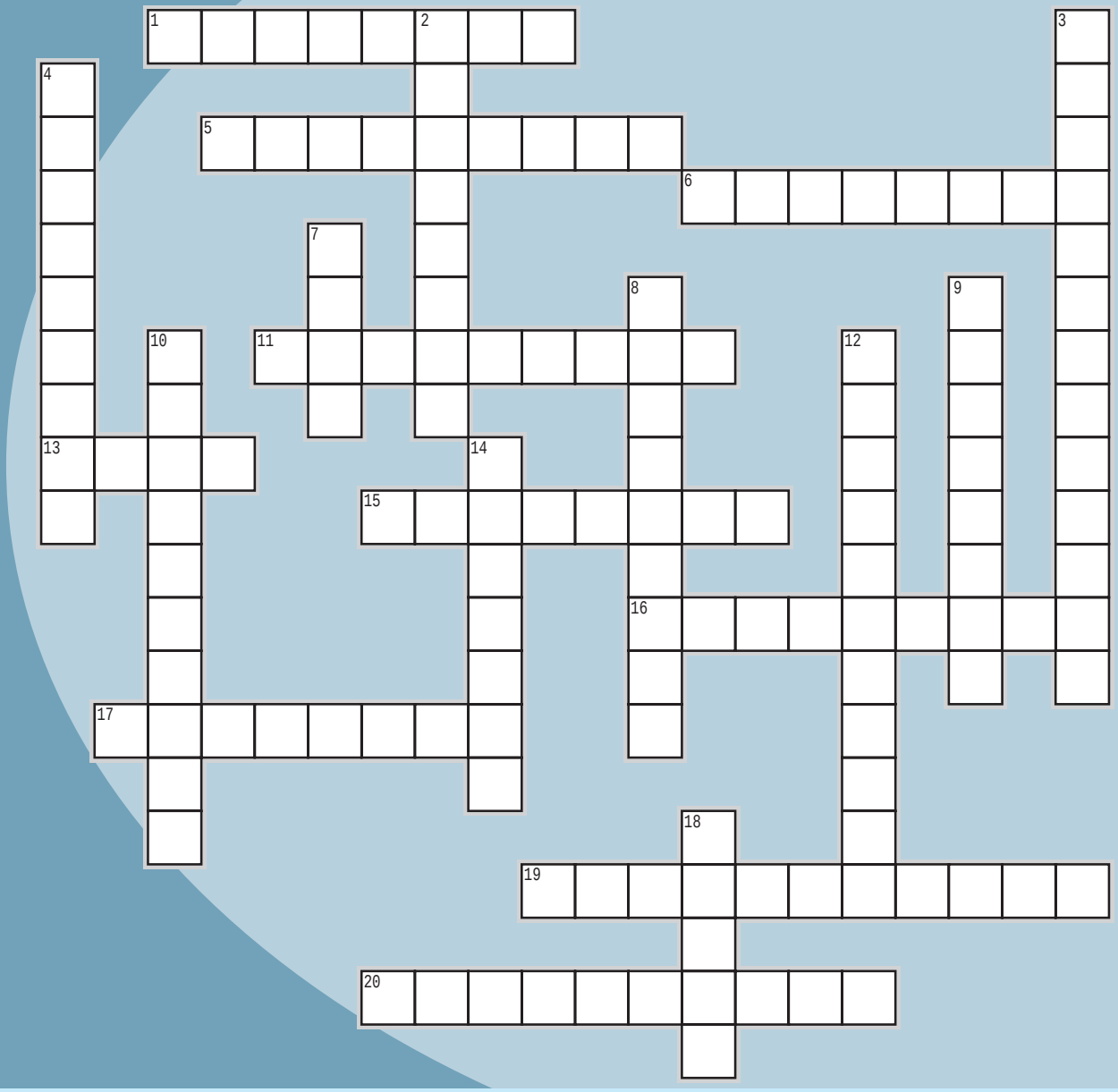


Free upgrade to 16 gigabyte USB during this promotion.



Limited time offer.
For details and to purchase visit:
www.cc-webshop.com

CROSSWORD



Across

1. Saves it for later
5. A capacitor that holds and releases a DC charge
6. Says yes to high frequencies and no to low frequencies [two words]
11. A circuit that does things on its own
13. An inductor
15. Defaults in the event of a power loss [two words]
16. Measures electric current
17. A resistor whose value decreases when its voltage gets too high
19. Combines a transmitter and a receiver
20. Created in the early '70s and replaced by CD-ROMs in the '90s [two words]

Down

2. 1,099,511,627,776 bytes
3. Separated signals [two words]
4. Used on PCBs instead of jumper blocks [two words]
7. Windows-based computers rely on it to get going
8. Circuits feeding one another
9. Transfers energy from circuit to circuit
10. A ring of wires [two words]
12. A layer of copper that helps reduce noise and increase potential [two words]
14. Electrical conductors that can be easily pulled apart [two words]
18. Works the opposite way as a cathode

The answers will be available in the next issue and at www.circuitcellar.com/crossword.

IDEA BOX

THE DIRECTORY OF PRODUCTS AND SERVICES

AD FORMAT: Advertisers must furnish digital files that meet our specifications (www.circuitcellar.com/advertise). ALL TEXT AND OTHER ELEMENTS MUST FIT WITHIN A 2" x 3" FORMAT. E-mail adcopy@circuitcellar.com with your file or send it to IDEA BOX, Circuit Cellar, 4 Park Street, Vernon, CT 06066. For current rates, deadlines, and more information contact Peter Wostrel at 800.454.3741, 978.281.7708 or peter@smmarketing.us.

The Vendor Directory at www.circuitcellar.com/vendor/ is your guide to a variety of engineering products and services.

JazzOS - Java for Makers

Get started with the JazzOS Explorer

Program in Java

32 Bit Native Code Performance

28 Multipurpose I/O Channels

8 Megabytes RAM
16 Megabytes FLASH

Plug-and-Play Development

Starting from $69.00



[PERNETICS] Practical Innovation

PERNETICS.COM
(800) 651-8698



Add a Touch Screen to Your Embedded Product

- No special OS or Library Required.
- Programming GUI is Simple.
- Development Kit = Up and Running in Days.

Learn more at reachtech.com, call 510-770-1417 or email info@reachtech.com.

REACH TECHNOLOGY INC. 20 1988-2008

Buy vs. Build? Download Handbook at reachtech.com.

Display with Touch Screen



- Embedded, Standalone, C/C++ Programmable...
- Low Cost User Interface with Ethernet, USB, CF, RS232, ADC, DAC, I/Os, Relays...
- U-Drive™ 5.7" QVGA TFT, CAN, Ethernet, CF, ADC, DAC, ...

start at $379 OEM

60+ Low Cost Controllers with TFT, ADC, DAC, UARTs, 300 I/Os, solenoid, relays, CompactFlash, LCD, Ethernet, USB, motion control. Custom board design. Save time and money.

TERN INC. 1950 5th Street, Davis, CA 95616 USA
Tel: 530-758-0180 • Fax: 530-758-0181
www.tern.com • sales@tern.com

VISA MasterCard AMERICAN EXPRESS

Artila
www.artila.com
sales@artila.com



M-506 Single Board Computer

Linux 2.6.38
ATMEL 400MHz ARM9 Processor
128MB SDRAM
128MB NAND Flash
Ethernet/Serial
USB/GPIO/SD Card
Supports LVDS/Up to 1280 x 800 LCD and touch screen

Contact
(West Coast)
Aaxeon Tech
1-877-AAXEON5

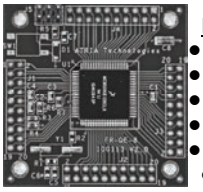
(East Coast)
WDL Systems
1-800-548-2319

BPS BusBoard Prototype Systems
Prototyping PC Boards
Many Patterns
BusBoard
StripBoard
PadBoard
ProtoBoard
ProtoBoard-2H
PowerBoard
SMTBoard
SMTpads
PC BreadBoards
See our site
www.BusBoard.us
Available From
JAMECO ELECTRONICS
MOUSER ELECTRONICS
a tti company
amazon.com
amazon.co.uk

ANALOG and DIGITAL for PC/104
GPIO-104
8 ANALOG INPUTS
4 ANALOG OUTPUTS
24 DIGITAL I/O
Need More Analog I/O?
ADIO-104
16 ANALOG INPUTS
8 ANALOG OUTPUTS
24 DIGITAL I/O
PULSE ACCUMULATOR
SCIDYNE®
Also Available
•96 DIGITAL I/O
•24 HIGH POWER I/O
•RELAY/OPTO I/O
•EXTERNAL EXPANSION
•PC/104 PROTOTYPING
1-877-SCIDYNE
info@scidyne.com
www.scidyne.com

Disk On Chip
Ready for Delivery
From 16MB to 128MB Available!
Call 530-297-6073 Sales@jkmicro.com
www.jkmicro.com
JK microsystems, Inc.
International Orders Welcome

NEW APP! Download at iTunes
CC Electronic Toolbox
For iPad, iPhone, & iPod Touch
Specially designed for electronic engineers, this app provides easy access to essential electrical information and multiple conversion tools!
BJT Calculations, Ohm's Law calculator, Type listings for Resistors including SMD types and Capacitors, database of common electrical components, NE555 Circuit design, CC Tech Guide for authors & so much more!
www.circuitcellar.com/toolbox



Microcontrollers
- MCF51AC256
- MCF51CN128
- MCF51JM128
- MCF51QE128
- MC9S08AC128
- MC9S08JM64
- MC9S08QE128

Modules
- Bluetooth
- RS-232
- USB
- RF
- 2 x 20 LCD
- 4 x 4 Keypad
- Motor Control
- Real Time Clock
- Power Supply

GREAT For:
- Projects
- Breadboards
- Experiments
- Labs
- Prototypes

Kits Available

Learn about BASIC ON BOARD

ATRIA *Technologies Inc.*

www.AtriaTechnologies.com

RF Specialists

RF Modules

GSM/GPRS

Bluetooth

Data Loggers

GPS Wi-Fi

ZigBee Pro

LEMOS INTERNATIONAL

www.lemosint.com

MaxSonar

Perfect for OEMs & Engineers

XL-MaxSonar-EZ
- High acoustic power
- Real-time calibration
- Tiny size
- $39.95 USD qty. 1

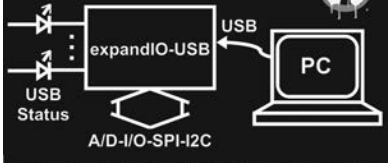
MaxSonar-WR (IP67)
- Industrial packaging
- Weather resistant
- Standard ¾" fitting
- $99.95 USD qty. 1

LV-MaxSonar-EZ
- Power-up calibration
- Smallest MaxSonar
- Low power, 2.5V-5.5V
- $29.95 USD qty. 1

MaxSonar-WRC (IP67)
- Compact packaging
- Weather resistant
- Quality narrow beam
- $99.95 USD qty. 1

www.maxbotix.com

Revolutionary new expandIO-USB *chip*



- Ideal for adding USB to sensors & peripherals
- No drivers needed for Windows, Mac, Linux
- No microcontroller programming required
- Also check out our USB-232 USB to UART

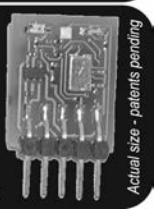
www.hexwax.com - Buy from Mouser & Farnell

Amazing PIC programmer

Most devices supported
ICSP, SQTP, & copy limits

$32 at Digikey & Mouser

www.flexipanel.com



Actual size - patents pending



$795 introductory price

- ¼ " Ultrasonic Microphone & Amplifier
- 500 Hz – 100KHz +/- 3 dB BSL
- Sensitivity: 1 V/P

KarlZurn@gmail.com
321-613-3862

FLORIDA RESEARCH INSTRUMENTS, INC.

www.FloridaResearchInstruments.com

i2cStick™



Discover I²C

Read, Write, Connect...

MCC Micro Computer Control

www.mcc-us.com

microEngineering Labs, Inc.

www.melabs.com 888-316-1753

The industry-standard BASIC compiler for Microchip PIC® microcontrollers.

PICBASIC PRO™ Compiler 3.0



Experimenter - $49.95
Silver Edition - $119.95
Gold Edition - $269.95

Academic Licensing Available

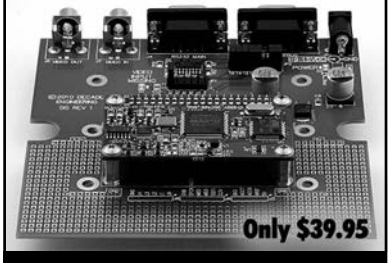
Download a FREE trial version now.

www.PBP3.com

PICBASIC and PICBASIC PRO are trademarks of Microchip Technology Inc. in the USA and other countries.
PIC is a registered trademark of Microchip Technology Inc. in the USA and other countries.

NEW SIS Board

Turbocharges the development of low-cost modular video text and graphics generators using **BOB-4**!

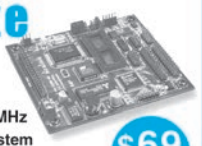


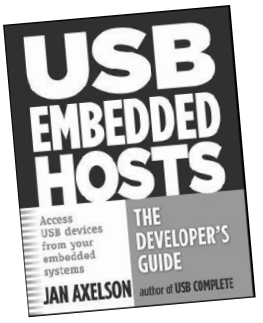
www.decadenet.com

DECADE ENGINEERING
503-743-3194 Turner, OR, USA

Ideal for Sensor Interfacing
PIC18F45K22 Development Kit
· Features 10 Analog Inputs - Ideal for Sensors
· ADC, Temperature Sensing Tutorials Reduce Learning Curve
· Built-in Compiler Functions for A/D, Sensor Drivers Speed Development
Kits start at $125

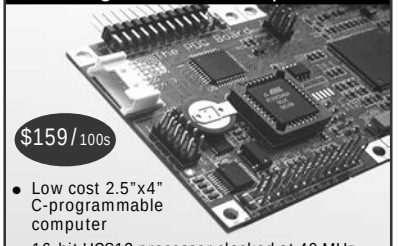

PIC18F45K22 Development Kit with Programmer & Tutorial
www.ccsinfo.com/CCJAN
sales@ccsinfo.com
262.522.6500 x35
Expires: January 31, 2012
CCS

Flashlite 186

• 186 processor @ 33 MHz
• DOS w/ Flash File system
• 44 Digital I/O lines w/ CPLD
• Console / Debug Serial Port
• 7-34V DC or 5V DC power • 2 Serial Ports
• Accepts 8MB DiskOnChip • 2 16-bit Timers
• 512K DRAM & 512K Flash • Watchdog Timer
• Expansion options with Peripheral Boards
$69 QTY 1
$99 Development System
Development kit includes:
• Flashlite 186 controller
• Borland C/C++ ver 4.52
• FREE Email Tech Support
• Serial Driver library
• AC Adapter and cable
Call 530-297-6073 Email sales@jkmicro.com
On the web at www.jkmicro.com
JK microsystems

Access USB Devices from your Embedded Systems


USB Embedded Hosts
The Developer's Guide
Jan Axelson
$29.95 LVR.COM
From the author of *USB Complete*

Connect With Design Engineers From Around The Globe.
Reserve advertising space in Circuit Cellar and CC News Notes today!
Strategic Media Marketing
978.281.7708
peter@smmarketing.us
www.smmarketing.us

PDQ Board™ - A Fast I/O-Rich Single Board Computer

$159/100s
• Low cost 2.5"x4" C-programmable computer
• 16-bit HCS12 processor clocked at 40 MHz
• 8 PWM, 8 counter/timer, and 8 digital I/O
• 16 10-bit A/D inputs
• Dual RS232/485 ports, SPI and I²C ports
• 512K on-chip Flash, 512K RAM with Flash backup
• Plug-in I/O expansion, including Ethernet, Wi-Fi, GPS, 24-bit data acquisition, UART, USB, Compact Flash card, relays, and more ...
Mosaic Industries Inc.
tel: 510-790-1255 fax: 510-790-0925
www.mosaic-industries.com

ALL ELECTRONICS CORPORATION
Electronic and Electro-mechanical Devices, Parts and Supplies.
Wall Transformers, Alarms, Fuses, Relays, Opto Electronics, Knobs, Video Accessories, Sirens, Solder Accessories, Motors, Heat Sinks, Terminal Strips, L.E.D.S., Displays, Fans, Solar Cells, Buzzers, Batteries, Magnets, Cameras, Panel Meters, Switches, Speakers, Peltier Devices, and much more....
www.allelectronics.com
Free 96 page catalog
1-800-826-5432

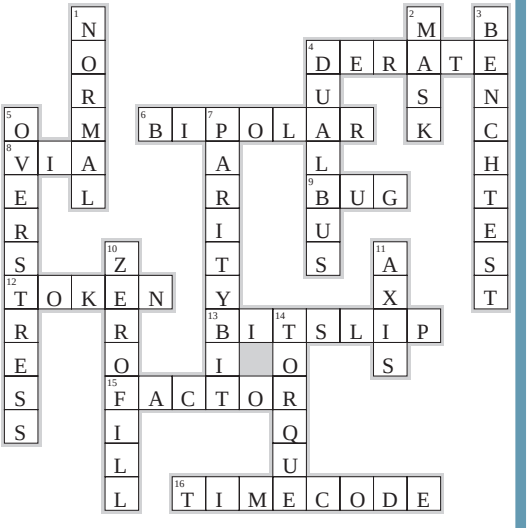
CROSSWORD ANSWERS from Issue 257

Across

4. **DERATE**—Increase reliability by reducing classification
6. **BIPOLAR**—Having not just one polarity
8. **VIA**—A hole in a board
9. **BUG**—Error
12. **TOKEN**—Offers authentication
13. **BITSLIP**—What happens when you've lost a bit [two words]
15. **FACTOR**—When combined, more than one forms a product
16. **TIMECODE**—Used for digitial synchronization [two words]

Down

1. **NORMAL**—AKA, "the perpendicular"
2. **MASK**—Use one data word to alter another
3. **BENCHTEST**—A lab analysis simulating an actual I/O [two words]
4. **DUALBUS**—Opposing buses [two words]
5. **OVERSTRESS**—Caused by too much voltage or current
7. **PARITYBIT**—Can make one odd [two words]
10. **ZEROFILL**—000000, for example
11. **AXIS**—Right at the heart of things
14. **TORQUE**—Tau symbol



INDEX OF ADVERTISERS

Page		Page		Page		Page	
59	2012 International CES	55	DesignCon 2012	71	Humandata Ltd.	7	Mouser Electronics, Inc.
78	All Electronics Corp.	9, 12	DesignSpark chipKIT™ Challenge	21	Ironwood Electronics	C2	NetBurner
64	AP Circuits	21	egnite GmbH	19	Jameco Electronics	C4	Oscium
63	APEC 2012 Conference	34, 35	Elektor	17	Jeffrey Kerr, LLC	75	Pernetics
52	APEX Expo 2012	67	Elektor	13, 33	JK microsystems, Inc.	53	Pololu Corp.
45	ARM	57	Elprotronic, Inc.	76, 78	JK microsystems, Inc.	75	Reach Technology, Inc.
75	Artila Electronics Co.	11	Elsevier, Inc.	27	Labcenter Electronics	41	Saelig Co., Inc.
77	Atria Technologies Inc.	64	EMAC, Inc.	78	Lakeview Research	76	Scidyne
23	Bitscope Designs	57	SiliconRay	33, 77	Lemos International Co.	2, 3	Technologic Systems
76	Busboard Prototype Systems	77	FlexiPanel Ltd.	77	Maxbotix, Inc.	75	Tern, Inc.
73	CC Gold	77	Florida Research Instruments, Inc.	77	MCC, Micro Computer Control	43	Triangle Research, Inc.
31	Comfile Technology, Inc.	29	Front Panel Express LLC	77	microEngineering Labs, Inc.	29	Tronicszone
78	Custom Computer Services, Inc.	C3	FTDI Chip	66	Midwest Patent Law	5	Wiznet
77	Decade Engineering	13	Grid Connect, Inc.	78	Mosaic Industries, Inc.		

PREVIEW of February Issue 259

Theme: Wireless Communications

QRSS-Rx: A Network for Radio Communication

KartTracker: GPS-Based Vehicle Timer

NakedCPU (Part 1): A Roadmap for Navigating Hardware Documentation

EMBEDDED IN THIN SLICES **Getting Started with Embedded Linux (Part 3):** Linux Licensing Issues

THE CONSUMMATE ENGINEER **Build a Robust Power Supply (Part 2):** Abnormal Surge Voltage

FROM THE BENCH **Multi-Serial Protocol Tool**

ABOVE THE GROUND PLANE **RFI Bypassing (Part 1):** Capacitor Self-Resonance

THE DARKER SIDE **Playing with High-Speed ADCs**

ATTENTION ADVERTISERS

March Issue 260 Deadlines

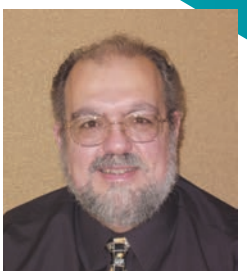
Space Close: Jan. 13
Materials Close: Jan. 20

Theme
Robotics

Bonus Distribution
ESC Silicon Valley; Trinity College Robotics Contest

Call Peter Wostrel
now to reserve your space!
800.454.3741 or 978.281.7708
e-mail: peter@smmarketing.us

PRIORITY INTERRUPT



by Steve Ciarcia, Founder and Editorial Director

Too Many Layers

It seems like the smarter my PCs, smartphones, and DVRs get, the dumber I feel. Obviously, it takes more than being an EE these days to understand (or translate) the directions on the average tech device. I'm beginning to feel that if I didn't design it from scratch, I haven't a prayer figuring out what's under the covers. Forget comprehending how these exceedingly complex devices really work, and gone are the days where we could pop the top and reconfigure them.

The problem is, I'm beginning to think designers don't care how complex these systems are or that managing their complexity is somehow on autopilot (robots building the robots). Even if I did want to take my new 16-channel DVR apart, what can I do with it? When you start digging in, all you encounter is an astounding number of additional technologies all stacked on top of each other.

I don't want to complain that "The World's Source for Embedded Electronics Engineering Information" is pushing its creator off the cliff. But if I am feeling intimidated by the complexity of some of the newest electronics, how is the rest of the world handling it? Is it simply an age thing, or is there is a new younger culture with a blind faith in everything "smart" and I'm going in kicking and screaming?

"Faith" for me comes from understanding how something works. It wasn't all that long ago that we could understand all of the electronic devices in our lives at a fairly deep level: AM and FM radio, record players, tape recorders, digital clocks, landline telephones, and analog televisions. When digital logic was developed, it also wasn't that bad keeping up. A lot of interesting things could be built with SSI/MSI logic and microprocessors. You or I could design and build a Z80 computer, write the BIOS for it, and use it for useful tasks with an off-the-shelf operating system or Assembly code and C. If we wanted to "improve" a commercial design using a micro (OK, hack it), we pretty much knew what was what on the circuit board, and where to stick a scope probe, and so on.

Fast-forward to today. Look at your latest HDTV, Internet video box, or smartphone. Got any clues about what the operating system is or even dare to open the box? Heaven forbid you enter a command incorrectly. You might end up ordering a movie for every resident in Seattle or sending your phone address list to everyone in Germany. The combination of all these operating systems, programs, and design layers is so complex I'm not sure who really understands all of it.

My concern is related to the way the layers of technology build up over time, obscuring the basic concepts and making the levels to entry for new people that much higher. Take a look at current home entertainment systems. At the very least, they include an HDTV (possibly with Internet hookup), a cable box, a DVD player, and a DVR. The interfaces include F connectors for the RF, HDMI connectors for digital video, and RCA connectors for legacy analog video and audio. It was hard enough for some people to learn to use a remote control. Now they have to become IT specialists to watch their own TVs.

Seriously, does designing extremely complicated electronics have to mean there is always an extremely complicated learning curve to using them? What gives with consumer electronics? Haven't they learned anything from the automobile industry? Cars are the highest-density computer electronics we own, but using them is a whole lot less in your face than configuring an HDTV theater setup. If you could start a car and drive in 1960, you can get in and push the button or turn the key and make one go today. If you knew how to put your '91 Caddy in cruise control, you can do it on a 2012 just as easily. Yeah, now it's active cruise that keeps you a constant distance from the car ahead, but you didn't have to learn to program or get an EE degree because they changed from analog to digital and improved the product. Because car companies know there are fundamental concepts and ingredients in their products, some subset of knowledge and experience will always enable you to drive any car.

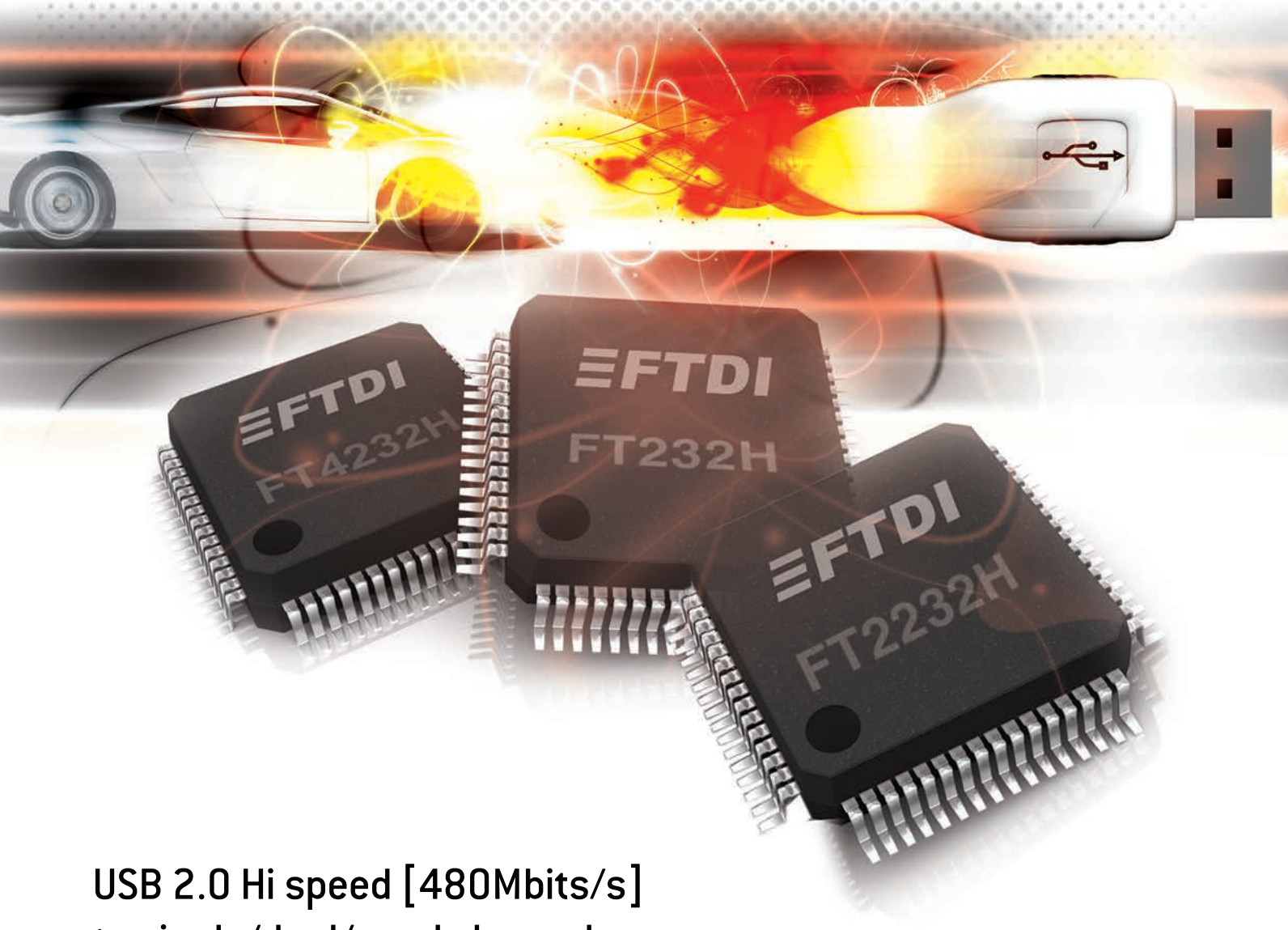
Cars aside, I have nagging doubts about younger generations being able to internalize and make use of these fundamental concepts the way my generation does, and I'm concerned about the "brain drain" as older, more disciplined engineers give way to younger ones who have a "smart-everything" mentality. I may be overly alarmed, it seems like the only electronic solution from a graduate engineer these days is slapping a microprocessor, FPGA, or ASIC on it. The development tools are complex, you're toast if you don't do C programming, and with them there's no way to "think small."

I'm not saying it won't work. As a person who grew up alongside all this technology, I embrace its success completing any task. However, it seems like the new culture doesn't even bother to look for other options or, heaven forbid, an analog solution. Being a good engineer has to include more. Medical schools teach the importance of bedside manner in creating great doctors. Perhaps a little engineering version of "bedside manner" needs to be taught to engineers so they can empathize better when it comes to using all this stuff.

steve.ciarcia@circuitcellar.com



USB IN THE FAST LANE



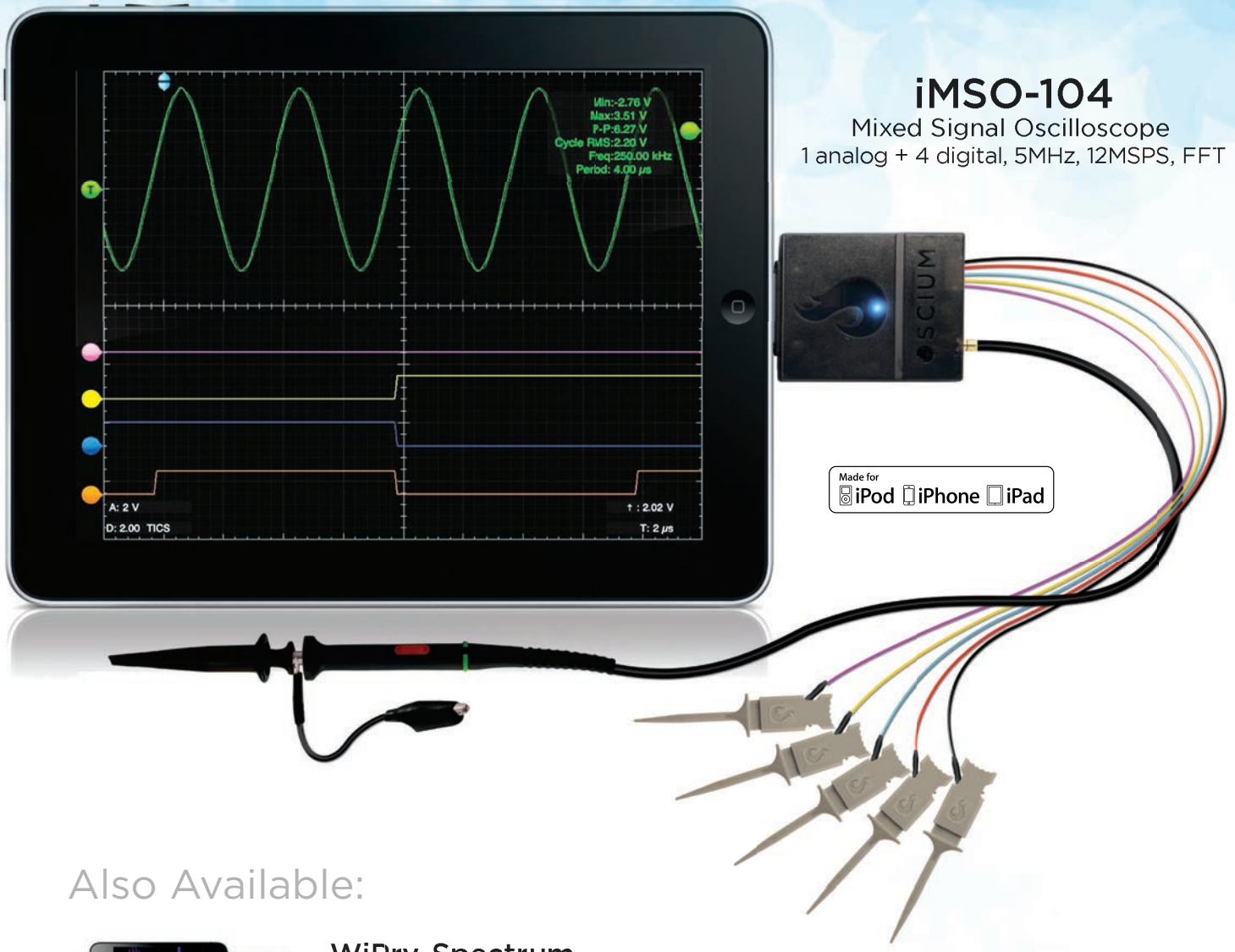
USB 2.0 Hi speed [480Mbits/s]
to single/dual/quad channel
independent configurable parallel/serial interface
UART/JTAG/SPI/I^2C & parallel FIFO

FTDI Chip

USB MADE EASY

www.ftdichip.com

Tired of **Buttons** & **Knobs**?

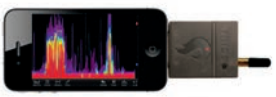


iMSO-104

Mixed Signal Oscilloscope

1 analog + 4 digital, 5MHz, 12MSPS, FFT

Also Available:



WiPry-Spectrum

2.400-2.495GHz, -40dBm to +20dBm

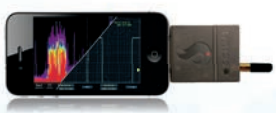
Resolution 2.0dBm, Sweep Time 200ms



WiPry-Power

100MHz to 2.7GHz, -45dBm to +20dBm

Resolution 0.2dBm, 2 µs/div to 1 s/div



WiPry-Combo

WiPry-Spectrum & WiPry-Power Combined!

OSCIUM

www.oscium.com