

CIRCUIT CELLAR

THE MAGAZINE FOR COMPUTER APPLICATIONS

September 2010
Issue 242

DATA ACQUISITION

Data Transmission, Gathering,
and Decoding

Robust Design, Uncertainty
Management

How to Build a Precision
Temperature Controller

A Reexamination of State
Machines

MCU + FPGA "Fusion"

PLUS
INTERDISCIPLINARY ENGINEERING

An Academic Uses His Electrical Engineering
Skills for Application Development in Fields
Ranging from Neurobiology to Computer Science



www.circuitcellar.com



Design Inspiration

At *Circuit Cellar*, we pride ourselves on providing readers with a variety of useful content, much of which I've described here in the past: the highest-quality project articles, need-to-know information about cutting-edge technology, helpful design/programming tips, and more. You've come to expect such things from us. But all the information, tools, and technical know-how in the world mean nothing if you aren't consistently inspired to hit the workbench each day, rethink every design problem, and fight through long bouts of designer's block.

This month, we bring you the third interview in what we plan to be a long list of interviews to come. Our goal has been to present you with insight into what interests and drives some of your most successful peers, with the hope of inspiring you to work harder than ever to achieve your occupational aspirations and design ambitions. This month you learn about an academic who seems to have so much building, researching, teaching, and developing going on that the entire *Circuit Cellar* staff is frankly baffled that he has time for anything unrelated to design (or programming, or physics, or graphics, or neurobiology, or chemistry). Bruce Land is an inspiration to anyone with an interest in embedded development, programming, or electronics in general (p. 18). In addition to his duties as an instructor at Cornell University, Bruce uses his many talents to work on applications in the fields of computer engineering, neurobiology, chemical kinetics, and beyond.

I'm sure Bruce's interview will inspire you to get right to work. But before you drop this magazine and start designing, read George Novacek's column on page 22. George is our go-to author on the topic of robust design. His tips will make you a more well-rounded engineer.

On page 26, David Ludington describes a project for building precision temperature control circuitry. You can apply the concepts to bring more control to virtually any temperature system you're designing.

Have you heard of the Village Telco project? Its purpose is to develop an affordable alternative to mobile phones. Check out David Rowe's mesh telephony design on page 36.

Turn to page 44 if you're ready to start working on a project for the NXP mbed Design Challenge 2010. Clemens Valens gets you started with an introduction to mbed and a description of how he built a scrolling LED message board.

On page 50, columnist George Martin revisits the important topic of state machines. It's a good review for anyone planning a complex application.

Interested in data transmission and decoding? On page 56, Jeff Bachiochi describes how to design a keyfob decoder.

Tom Cantrell wraps up the issue with an article about a fascinating idea that isn't exactly new: an MCU and FPGA on the same chip (p. 64). Tom investigates whether Actel's current offering fits the bill.

cj@circuitcellar.com



CIRCUIT CELLAR®

THE MAGAZINE FOR COMPUTER APPLICATIONS

FOUNDER/EDITORIAL DIRECTOR

Steve Ciarcia

PUBLISHER

Hugo Vanhaecke

EDITOR-IN-CHIEF

C. J. Abate

ASSOCIATE PUBLISHER

Shannon Barraclough

WEST COAST EDITOR

Tom Cantrell

CUSTOMER SERVICE

Debbie Lavoie

CONTRIBUTING EDITORS

Jeff Bachiochi

Robert Lacoste

George Martin

Ed Nisley

CONTROLLER

Jeff Yanco

ART DIRECTOR

KC Prescott

NEW PRODUCTS EDITOR

John Gorsky

GRAPHIC DESIGNERS

Grace Chen

Carey Penney

PROJECT EDITORS

Ken Davidson

David Tweed

STAFF ENGINEER

John Gorsky

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

Valerie Luster

E-mail: val.luster@circuitcellar.com

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

PRINTED IN THE UNITED STATES

CONTACTS

SUBSCRIPTIONS

Information: www.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®, THE MAGAZINE FOR COMPUTER APPLICATIONS (ISSN 1528-0608) is published monthly by Circuit Cellar Incorporated, 4 Park Street, Vernon, CT 06066. Periodical rates paid at Vernon, CT and additional offices. **One-year (12 issues) subscription rate USA and possessions \$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 © 2010 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.

INSIDE ISSUE

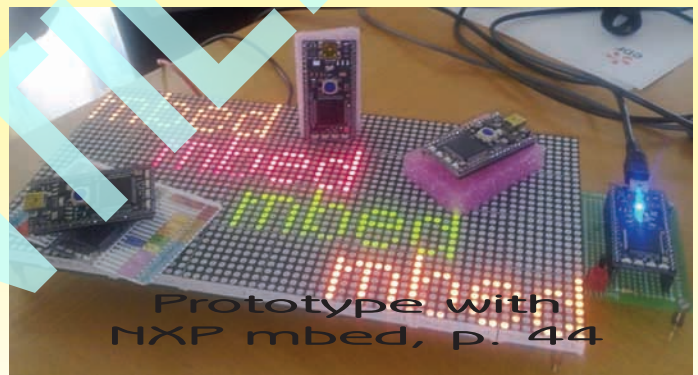
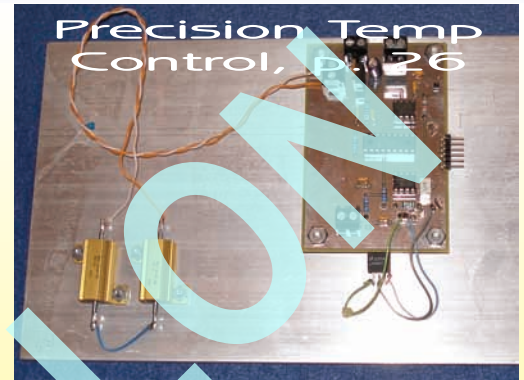
242

September 2010 • Data Acquisition

26 Precision Temperature Control Circuitry
David Ludington

36 Mesh Telephony System
The Mesh Potato Project from an Embedded Designer's POV
David Rowe

44 NXP mbed Design Challenge 2010 Primer
Rapid Prototyping
Build a Scrolling LED Message Board with an mbed/NXP LPC Platform
Clemens Valens



22 **THE CONSUMMATE ENGINEER**
Is the Door Closed?
Why Every Safety-Critical Decision Matters
George Novacek

50 **LESSONS FROM THE TRENCHES**
State Machines Revisited
Real-World Word Problems
George Martin

56 **FROM THE BENCH**
Transmit and Decode Data
Design and Implement a Keyfob Decoder
Jeff Bachiochi

64 **SILICON UPDATE**
Once More, With Feeling
An MCU + FPGA Without the Compromises
Tom Cantrell

TASK MANAGER 4
Design Inspiration
C. J. Abate

QUESTIONS & ANSWERS 18
Interdisciplinary Engineering & Research
An Interview with Bruce Land
C. J. Abate

NEW PRODUCT NEWS 8

TEST YOUR EQ 17

CROSSWORD 74

INDEX OF ADVERTISERS 79
October Preview

PRIORITY INTERRUPT 80
Mobile Metamorphosis
Steve Ciarcia

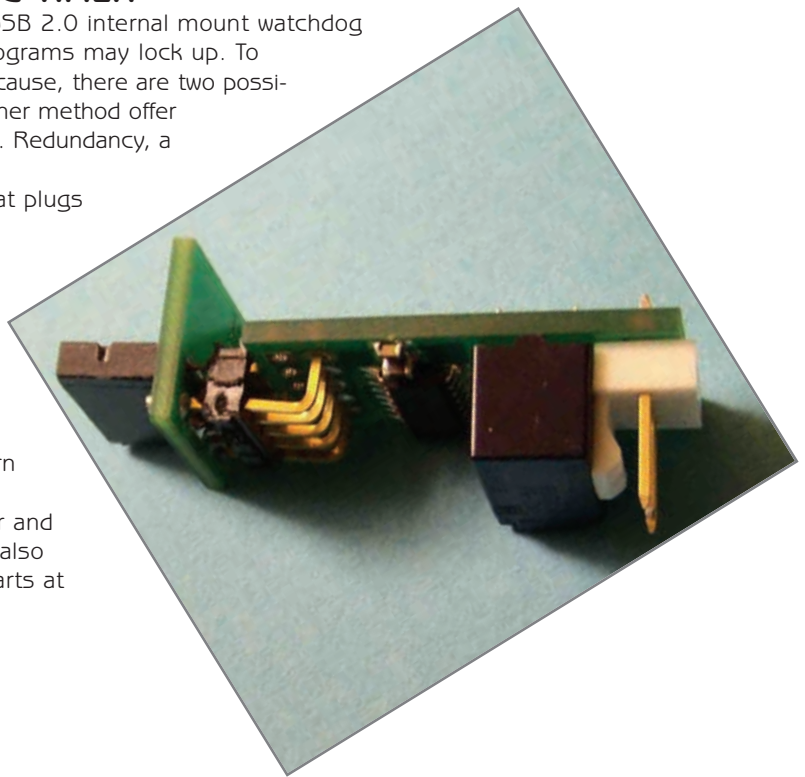
USB INTERNAL MOUNT WATCHDOG TIMER

J-Works is now offering the **WDT205 series** of USB 2.0 internal mount watchdog timer modules. Computer hardware devices or programs may lock up. To reduce the risk and possible damage a lockup can cause, there are two possible methods: redundancy or a watchdog timer. Neither method offer 100% protection, but both reduce the risk of failure. Redundancy, a duplication of hardware, is very expensive.

The WDT205 watchdog timers are a solution that plugs into an internal USB port meeting the Intel "Front Panel I/O Connectivity Design Guide" for a USB port. The host computer or application must communicate with the WDT205 by sending a message over the USB port. If the communication does not occur within that programmed time frame, the WDT205 triggers a relay to close for a programmable amount of time. This relay event can be used to trigger a warning device, turn off critical hardware or perform a system reset.

The software interface includes a windows driver and a DLL or Class Library API. A Linux support disk is also available. Single unit price for the WDT module starts at \$29.

J-Works, Inc.
www.j-works.com



UPDATED THREE-COMPONENT TRIAXIAL FORCE LINKS

Enhancements have now been implemented in the three-component, triaxial force link **93x7C** product range. Changes include expanded measurement capabilities from ± 1.8 k to ± 33.7 k lbf and other design enhancements for greater measurement flexibility and performance. These three-component dynamic force links are compact, fully calibrated, and preloaded for reliable high-precision dynamic force measurements in three orthogonal directions, in both tension and compression modes, across a variety of applications, regardless of the acting point of the force, with a large useable frequency range. All dynamic sensing elements are housed in a rust-free, sealed stainless steel housing.

General product line improvements to the three-component triaxial force link models include improved resolution and an expanded operating temperature range of -40° to 248° F, reduced hysteresis, an optional IP67 rated environmental sealing and plug connection via highly robust V3 multi-pole connector, which replaces the legacy three 10-32 negative connection. The newer versions are drop-in replacements for legacy models, and use the same cable type as those with types 93x7C already introduced. Specific enhancements to the force link include a more rigid construction and expanded measurement range of ± 0.8 k lbf across the x and y axes (previously ± 0.6 k lbf) and ± 1.8 k lbf across the z-axis (previously ± 1.1 k lbf).

Contact Kistler for pricing.

Kistler Instrument Corp.
www.kistler.com



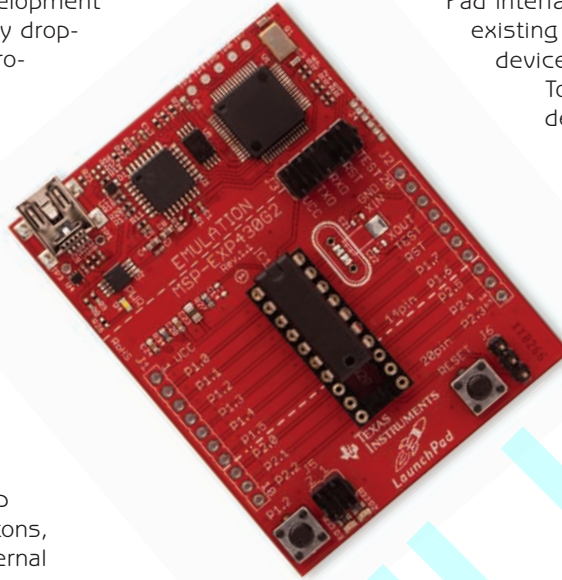
NEW PRODUCT NEWS

Edited by John Gorsky

MSP430 LAUNCHPAD DEVELOPMENT KIT

Texas Instruments has announced its new MSP430 MCU Value Line **LaunchPad** development kit. The open-source kit includes all of the hardware and software needed to easily launch designs based on the MSP430 Value Line MCUs. LaunchPad supports rapid prototyping and development by allowing developers to quickly drop-in MSP430 MCUs to evaluate, program, or debug devices. This flexibility allows developers to remove programmed devices to be placed on a custom board, or remain plugged into LaunchPad to leverage on-board buttons, LEDs, and breakout pins for external components.

Key features and benefits of the development kit are a DIP target socket, which supports up to 20-pin devices, and pushbuttons, LEDs, and breakout pins for external



components to allow the kit to function as a stand-alone system. Also featured is an integrated, USB-powered emulator that permits flash programming, firmware debugging, and serial communication and eliminates need for external emulator. The LaunchPad interfaces with any MSP430 Value Line MCU, existing e2430 target boards and MSP430 devices with Spy Bi-Wire capabilities.

To ease development, free compilers and debuggers are available for download including Code Composer Studio IDE and IAR Embedded Workbench. The kit comes with two MSP430

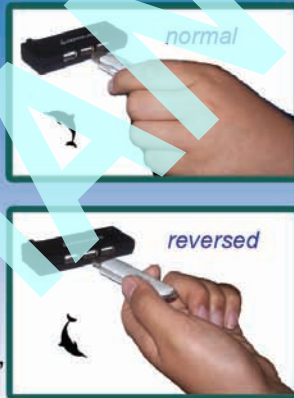
devices. One is preprogrammed with demo firmware to demonstrate the use of on-chip peripherals, including 10-bit ADC, comparators, and internal temperature sensor.

The LaunchPad costs \$4.30 in single quantities.

Texas Instruments
www.ti.com

Why Flipper™ USB?

- **Fast, flip-able, reversible,** “push-in” in either orientation
- **Easy,** no looking, no trying, no thinking, no guessing...
- **Friendly,** won't damage computers & USB devices
- **Safe,** spare connection is provided
- **Affordable, Tested, Certified, Patented & Trademarked!!!**



USER-FRIENDLY USB CONNECTOR

The new Flipper USB products are unlike conventional USB products as the patented connectors can plug into computers in either orientation. Flipper USB has many advantages over the traditional non-“flip-able” USB devices. Flipper USB is user-friendly and safe. With Flipper USB, it fits into the computer port either way. The user simply plugs in a USB device without looking, trying, thinking or guessing. It just works. This could save the user tremendous time and frustration. Furthermore, since Flipper is “flip-able,” the computer users will never make the mistake of pushing the device too hard in the wrong orientation, therefore avoiding damage to the computer or the USB device.

In addition to the above “friendly-and-safe” features, the Flipper USB offers spare/back-up connections. In case of damage to one side of the connector, the user can simply “flip it over” and use the other side of the USB device without loss of data or connections. With all the features and advantages, Flipper USB products won't cost more, thanks to the advanced smart design which makes Flipper truly affordable.

UltraTek offers Flipper connectors for all types of USB-related products, such as flash disk, Wi-Fi, BlueTooth, mouse, keyboard, camera, cables, and adapters. In addition, the Flipper adapter converts the regular, non-Flipper USB product to a Flipper USB product.

An SMT USB-A connector costs \$0.65 in quantities of 100.

UltraTek
www.flipperUSB.com



Is the Door Closed?

Why Every Safety-Critical Decision Matters

Part of developing a “robust” system involves ensuring your design is safe and secure. Whether you’re building a small embedded app or a complicated door-opening control system for a plane’s landing gear bay, you must protect both your design and the end users by choosing the proper parts and design techniques.

It may be very important to know if the door is closed. The answer is obtained by acquiring data reflecting the door’s status. Every control system needs data to act upon. In embedded controllers, most of the time, the data is provided by various detectors and sensors. Whether the controller itself is analog or digital is a secondary matter. The primary consideration is the type of sensor and its characteristics in terms of the operating environment. Let’s consider an example.

There are numerous mechanical devices on an aircraft whose configuration you must know at any given time. Doors are such devices: landing gear bay doors, passenger doors, and so on. You must also know the configuration of the up and down locks for the landing gear. To obtain their status, you must use some kind of a sensor—a limit switch, for example—to detect the position of the device you’re monitoring. Then, the resulting signal is acquired by the controller, digitized (if necessary), and processed. The trick is to deliver that important signal to the controller consistently and reliably.

Mechanical limit switches—such as micro switches—are rarely seen on aircraft. They are not very reliable in the harsh aerospace environment, and their maintenance by replacement is costly. Instead, proximity detectors are commonly used to detect the positions of moving mechanical parts. This column is not a proximity detection tutorial, but you need to understand the technical basics of proximity

detection to follow my example. So, I’ll only briefly describe the principle of inductive proximity detectors as found in many real-world applications.

Figure 1 is a block diagram of a typical proximity detector. A proximity detector consists of an oscillator, a part of which is a sensing coil. As the metallic target attached to a mechanical part being monitored moves toward the coil, the coil impedance changes, and this results in a voltage or current change. That voltage or current change is decoded and the output driver is energized accordingly. The device is usually powered by two wires, and the current driving the device is monitored by the controller. This current is divided into five distinctive bands. Target far is usually a low-current region. Target near is usually a higher-current region. Both appear in Figure 1. Currents less than the minimum target far, higher than target near, or between the targets (noted in Figure 1) mean that the detector is not operating correctly and its data should be rejected. This way a proximity detector not only detects the position of the monitored component, but also indicates whether the device is functioning correctly—a crucial characteristic in safety-critical systems.

SWITCH OR SENSOR?

The proximity detectors come in two basic flavors. A “proximity switch” contains all the components in Figure 1 in one tiny package, interfacing with the controller with a twisted

wire pair. The controller needs only to monitor the current to derive the status. A “proximity sensor,” on the other hand, is merely the sensing coil, while the associated interface electronics are an integral part of the controller.

So which one of the two functionally identical devices should we use? The switch or the sensor? It depends. Make a wrong choice and you may regret it.

Proximity switches are simple to use and to design around. Plus, the total cost of implementation is low. Their interface bandwidth and impedance to the controller are low, so it’s easy to protect their signal from transients, electromagnetic interference (EMI), and all the nasty disturbances bouncing around the aircraft. Properly constructed, a proximity switch generates no measurable interference of its own.

The proximity sensor interface, on the other hand, is complicated. The sensing coil, far removed from the interface circuitry, must be connected by a shielded cable. It can generate unwanted emissions, the small reactance changes of the coil due to the target movement may be obliterated by noise and external interference. The system integration is time consuming and requires experience.

So, why would you even bother with the sensor? The proximity sensor has some important redeeming qualities. Only the sensing coil is exposed to the elements and not much bad can happen to a rugged coil in the tough environment of landing gear, inside a jet engine or a space shuttle. Heat it up to nearly a melting point, freeze it to cosmic cold, immerse it in water, shake it, hit it with a hammer—the coil will continue to work reliably. Try doing the same to the proximity switch! If there is a chance that the operating environment could unexpectedly exceed the relatively mild operating conditions of the proximity switch, you may be in for a costly controller redesign. A proximity switch on a passenger door in an aircraft’s inhabited environment will work just fine, but on control surfaces I’d want to see proximity sensors. I want to know that the thrust reverser doors, for example, are really retracted. Lives may depend on knowing this.

System design and its reliable data acquisition from its peripheral components require not just a thorough knowledge of the system and its operating environment, but also an understanding of the internal characteristics of the devices used to acquire the data. I used a simple, 1-data-bit example to illustrate the point, but the same consideration must be given to the selection of all systems’ peripheral devices.

FUNCTIONALITY MATTERS

It is easy to get carried away by the capabilities and sophistication of new technology, such as smart sensors, smart actuators, high-level languages, or distributed processing. One can declare a holy war on anything other than the most advanced methods, insisting that those and only those find their way into his design. Experienced engineers,

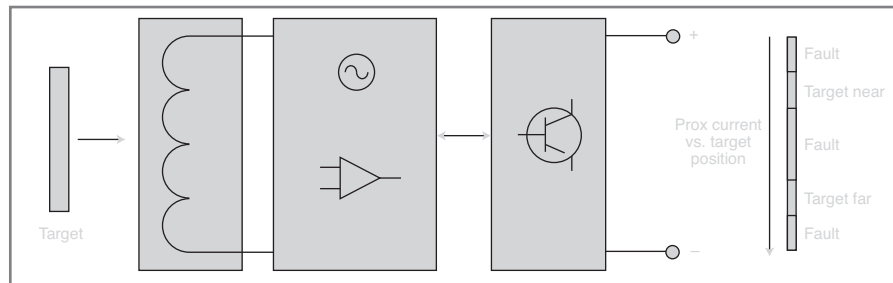


Figure 1—This is a typical proximity detector.

however, know that the essential part of engineering is a trade-off. Performance, reliability, operating environment, maintenance, cost, and time to market all play a role. Fad architectures and elegant “gee-whiz” designs mean little if they’re not 100% satisfactory. There is no universal best solution, not even for functionally identical tasks like monitoring a status of a door. A good engineer keeps his options open and is not afraid to use even old, time-tested design solutions if they best satisfy the issues at hand. Then and only then can he deliver optimal, robust designs. ☐

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 in Canada. George wrote 26 feature articles for Circuit Cellar between 1999 and 2004.

NEED-TO-KNOW INFO

Precision Temperature Control Circuitry

This precision temperature controller was designed for a calorimeter. The unique system controls the temperature of an outer metal box surrounding an inner calorimeter base plate. It maintains the box's temperature and controls temperature variations around the setpoint with an STD of less than 0.01°C.

There are many applications where temperature control is needed, such as home air conditioning and industrial processes. Of these applications, most do not require much precision. For example, in the home, the temperature is controlled to about 0.5° to 1°C, while in some industrial products electronic components are held to within 0.1°C. However, there are some measurement applications where much greater precision of temperature is required. One of these applications is a calorimeter which measures very small changes in heat to determine material or biological properties. The temperature surrounding the measurement cells must be held as constant as possible so that temperature changes in the environment do not corrupt the measurements.

In order to understand the level of temperature control that is possible, a multiphase study has been initiated. This study will seek to optimize the design of a calorimeter and will build and test the resulting temperature control circuits.

This article will describe the first phase of the study. A functional design of the calorimeter was performed and performance requirements

were generated. To get initial temperature data, the first hardware was built to control a metal plate inside a passive metal box. Results from testing this hardware will be used to optimize the temperature control loop.

In the next phase, the modified hardware will control the temperature of a metal box. This probably will require multiple control loops since there is not good thermal contact between the box top and the rest of the box.

In the third phase, an even more precise temperature control circuit will be needed to control the calorimeter base plate.

CALORIMETER BASICS

Figure 1 shows a system diagram for an isothermal (constant temperature) calorimeter. The calorimeter consists of a metal base plate at a fixed controlled temperature with two thermopile temperature sensors.

One thermopile measures the temperature difference between the measurement sample and the base plate. The other thermopile measures the temperature difference between the reference sample and the base plate. When a reaction in the measurement

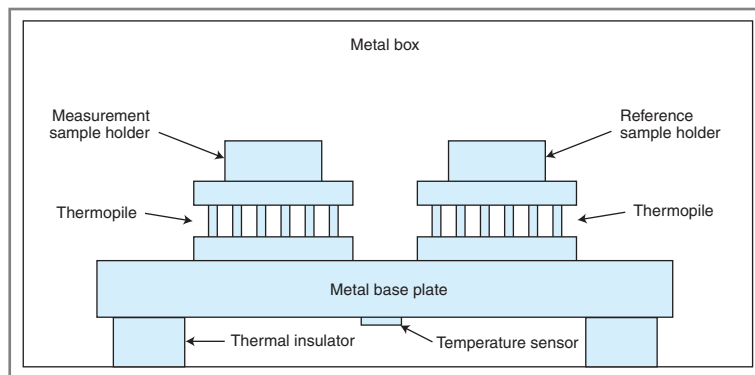


Figure 1—The heart of the calorimeter. Each thermopile outputs a voltage proportional to the temperature difference between the top and bottom plates of the thermopile.

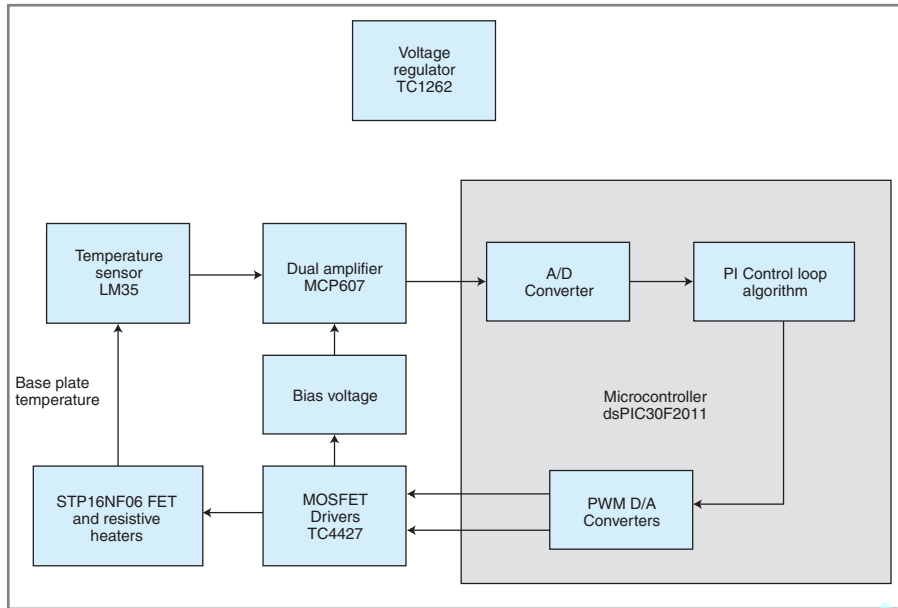


Figure 2—The control loop functional design. The dsPIC30F microcontroller has a 12-bit ADC and 16-bit PWM outputs. The thermal resistance and heat capacity of the heaters and plate filter the digital PWM signals to provide smooth temperature control of the plate.

sample results in a gain or loss of heat, the temperature of the sample is raised or lowered relative to the base plate temperature, and the thermopile generates a voltage signal that measures that difference. This is the signal you want. However, if the base plate temperature itself changes with time for any reason, this variation also becomes part of the measurement channel signal and cannot be

separated from the sample signal. The reference sample does not have internal heat generation. It has a zero output signal except for what's caused by temperature variations of the base plate. Thus, the signal from the reference channel signal can be used to provide compensation for the measurement sample by subtracting the reference signal from the measurement signal. This reduces the

effect of base plate temperature variations on the measurement sample signal. Of course, this only works for temperature variations that are common to both reference and measurement signals.

The thermopile temperature sensors are very sensitive and need the best low-noise amplifiers to measure the small voltage signals caused by sample temperature changes. For low-frequency bandwidths (time constant of 5 s), the voltage noise of these amplifiers can be reduced to approximately 1-nV RMS. This level of voltage noise corresponds to a temperature noise from the thermopiles of less than one-millionth of a degree Celsius or 1 μ K. (The Celsius and Kelvin temperature scales are the same except for an offset.)

Several things are required to preserve this level of sensitivity. First, care must be taken to minimize variations in the base plate temperature caused by external (to the calorimeter) environmental temperatures changes. In addition, care must be taken to ensure that any such changes equally affect both the measurement and reference samples so that the measurement signal can be compensated by the reference signal.

The isolation of the base plate is helped by putting a thermally conducting

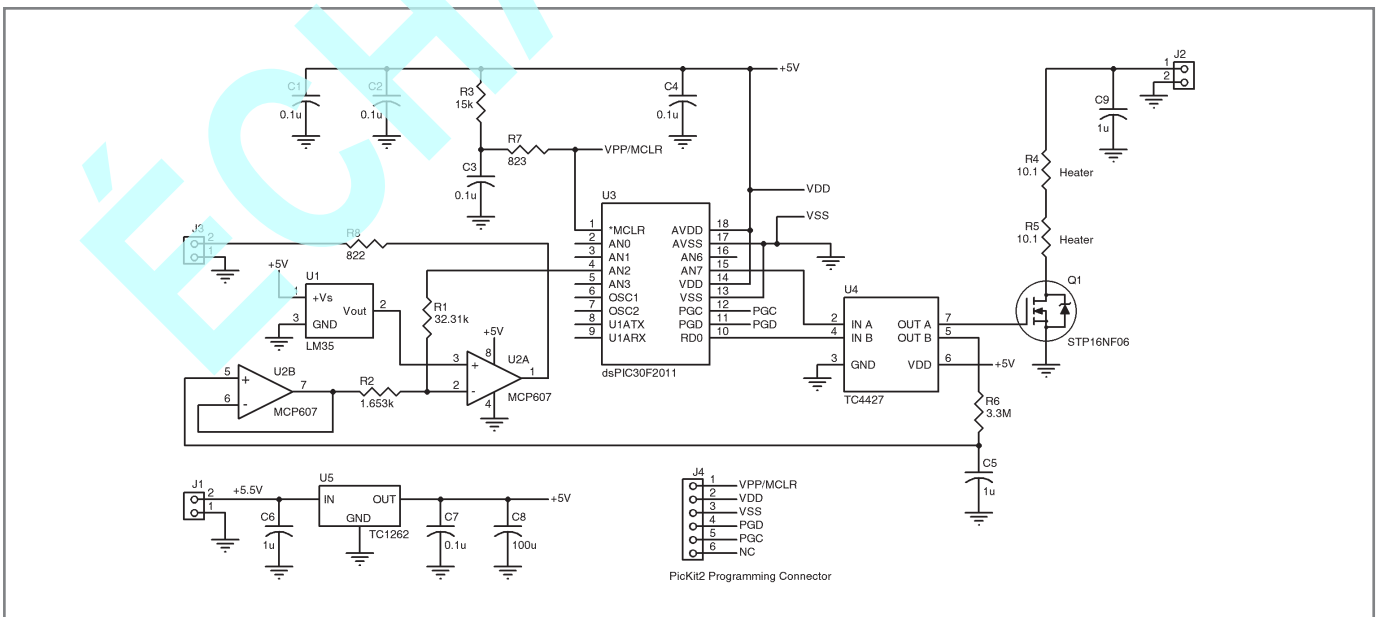


Figure 3—This is my generic controller design. The LM35 temperature sensor shown will be used to control the surrounding box temperature. This same controller will be used with a thermistor or RTD temperature sensor to control the base plate temperature.

Mesh Telephony System

The Mesh Potato Project from an Embedded Designer's POV

You might have heard about The Mesh Potato, a key component in The Village Telco project aimed at giving people in developing countries a low-cost alternative to mobile phones. This article focuses on the “embedded development” aspects of the project, such as the Wi-Fi SoC and interface design.

I am part of a team developing an open hardware mesh router with telephony called the Mesh Potato. Based on an Atheros Wi-Fi system on a chip (SoC), the Mesh Potato is a key component in the Village Telco—a project to give people in the developing world a low-cost alternative to mobile phones. The Mesh Potato is an 802.11bg router that you can plug a telephone into. It combines the functions of an Analog Telephony Adapter (ATA) and Wi-Fi router, and it includes BATMAN software for mesh routing and runs Asterisk for VoIP. In this article, I'll describe the Mesh Potato design (see [Photo 1](#)).

MESH NETWORKING

The Mesh Potato is a component in the Village Telco, which is a turnkey system for setting up small telephony networks in developing countries where cell phone call costs are prohibitive. The Village Telco project (including the Mesh Potato) has been funded by the Shuttleworth Foundation whose other projects include Ubuntu Linux. To mass-produce the Mesh Potato, we've partnered with Atcom, a VoIP hardware manufacturing company.

Sending a large number of voice calls over mesh networks is an interesting challenge. Wi-Fi protocols are optimized for sending large data packets reliably. However, voice packets are small and occasional packet loss is preferred over retransmission that introduces delay. In this article, I'll summarize what I've learned about VoIP over mesh networks.

Low-cost Wi-Fi SoC devices are highly integrated and very good at Wi-Fi, but they typically have few external interfaces. This article describes how we managed to

turn a router SoC with minimal interfaces into a mesh Wi-Fi telephony device.

Consider a simple Mesh Potato mesh network (see [Figure 1](#)). In mesh networks, each node relays messages for adjacent nodes. This extends the range of effective IP communications compared to infrastructure-mode networks that use a central access point. Mesh Potato A can't communicate directly with the gateway because it is out of range. However, nodes B and C can be used to relay packets for A. The BATMAN mesh routing algorithm used for the Village Telco automatically

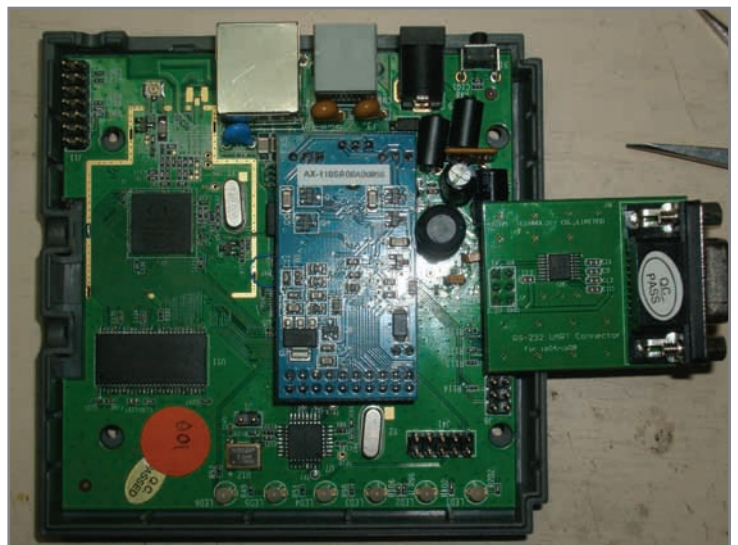


Photo 1—The Mesh Potato prototype has an FX5 module (center right) and a fitted R5-232 daughter board (far right). Along the top edge are Ethernet, telephone, and power connectors. On the left-hand edge is an Atheros AR2317 SoC and 16-MB SDRAM chip.

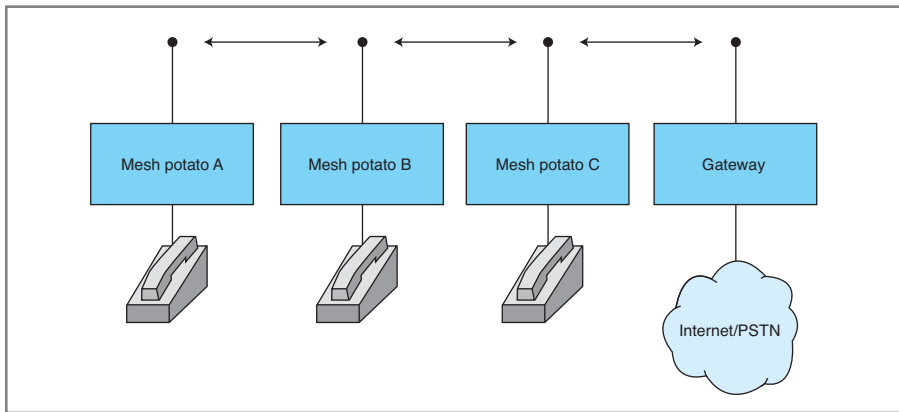


Figure 1—This is an overview of the Village Telco Mesh Network. The packets for a phone call from Mesh Potato A are relayed by Mesh Potatoes B and C to the gateway. The gateway connects to the public switched telephone network (PSTN), possibly via an Internet telephony service provider (ITSP).

determines the optimum route and adapts dynamically to changes in the Wi-Fi environment.

VoIP OVER MESH LIMITS

The 802.11bg Wi-Fi protocols add a lot of redundant information to the payload data. This is a minor problem with large (e.g., 1,500-byte) packets, as the overhead is amortized over a large number of data bytes. However, VoIP packets are small and must be sent at a high update rate (e.g., every 20 ms). The high packet rate generates significant overhead compared to larger packets. For example, a 33-byte GSM CODEC packet (20 ms of speech data) becomes an IP packet of 73 bytes once the RTP, UDP, and IP headers are included.

In addition to the overhead, there are

various per-packet timers and delays added by the protocol as part of the collision sense and retransmission mechanisms. So, after each packet is sent, the Wi-Fi protocol waits for ACKs and attempts to sense contention and other users on the radio channel. The physical layer synchronization mechanisms and MAC headers also consume radio airtime and bits. Sending a 73-byte data packet at 11 Mbps only takes 53 μ s; however, a total of 800 to 1,000 μ s is required to complete the protocol steps before another packet can be sent.

Mesh networks add another complication. All mesh routers occupy the same Wi-Fi channel. Three transmissions are required to route a packet from Mesh Potato A to the gateway above. This effectively reduces the

throughput of the channel by the number of hops (three in this case). Even though A can't communicate directly with C, they still tend to interfere with each other, and only one can transmit on the channel at any given time. This is because the interference range exceeds the communication range. Thus, the bandwidth of the mesh nodes tends to decrease linearly with the number of hops. In real-world mesh networks, this relationship is valid out to about five hops, after which interference from remote nodes tails off.

Just like Wi-Fi protocols, low-end Wi-Fi router CPUs are also more efficient with large packets. Every packet requires a cascade of interrupts and multiple function calls to be executed as the packets travel through the protocol stack. Thus, CPUs tend to be less efficient (and slower) at processing small VoIP packets. This overhead all occurs on a per-packet basis; therefore, the packet rate is the key constraint in designing VoIP-over-mesh Wi-Fi networks. Surprisingly, the speech CODEC and Wi-Fi channel bit rate have a fairly small effect on mesh capacity for VoIP traffic. This is because of the large per-packet overhead. A relatively small amount of time is spent actually transmitting speech CODEC data, so its bit rate is a minor factor.

Some examples show how small packets affect channel throughput. Sending 100-byte-payload packets

through a (single hop) 11-Mbps Wi-Fi channel resulted in a packet rate of 1,300 packets per second and payload throughput of 1.08 Mbps. In contrast, with 1,500-byte packets, the throughput was 5.5 Mbps. At 54 Mbps with 100-byte packets, we measured a payload throughput of just 2.8 Mbps!

Capacity can be improved by aggregating CODEC data. For example, packing four GSM CODEC frames in a single Wi-Fi packet, decreases the packet rate (and increases the

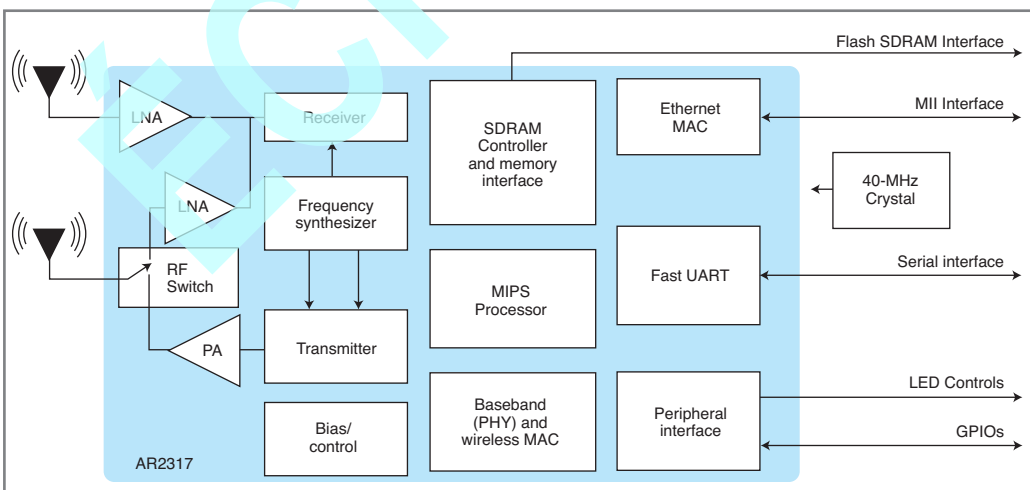


Figure 2—This is the Atheros AR2317 SoC. You have everything you need to build an 802.11bg Wi-Fi router, but how do you connect a FXS port? Our solution was to use the RS-232 serial port, some glue logic, and a heavily modified Linux serial driver.



State Machines Revisited

Real-World Word Problems

If you aren't using a real-time operating system (RTOS) but want to, take some time to revisit the topic of state machines. It's the perfect foundation for the complex applications you'll build in the near future.

All problems are word problems. Unfortunate, but true. I enjoyed math homework when the exercises involved problem solving and the problems were presented as a formula or a set of formulas. But soon we got to something like this:

John can paint a house in 3 days, and Bill can paint a house in 4 days. If John and Bill work together, how long will it take for them to paint a house?

The answer is not 7 days, but 1.7 days—that is, $1/(1/4 + 1/3)$.

When you talk to customers to get information about a problem, the results are a series of statements (word problems). One skill to develop is the ability to pull out the key facts or issues of the assignment from all the information.

Entry	Relative time	Action
23	0:05:23	Fireworks Bank 12
24	0:05:23	Balloon Drop Bag 3
25	0:05:23	Water Jet Pattern 3 Peak
26	0:06:12	Lights Pattern 7
27	0:07:19	Balloon Drop Bag 5
28	0:07:19	Water Jet Pattern 4 Peak

Table 1—A portion of the Music Time Track

The more you do this type of work the better you get at it.

REMEMBER STATE MACHINES

In my 2009 article titled "Embedded Breakup," I wrote about using state machines to help manage the system's resources and speed up programs (*Circuit Cellar* 230). You should dig out that issue for review. It is an excellent starting point for information about state machines. In *Circuit Cellar* 239, Ed Nisley described designing the software to interface and

Listing 1—Fireworks state machine first pass

```
// Operate the Music Time Line Interpreter.
while (NextCmd != DONE) {
    NextCmd = ReadNextCmd(); // Read the next command
    switch (NextCmd) {
        case NOP: {
            // Do nothing
        } break;
        case START_BALLOON_DROP: {
            // Send the command to start the balloons
        } break;
        case START_FIREWORKS: {
            // Send the command to shoot the fireworks
        } break;
        case START_WATER_JETS: {
            // Send the command to release the water jets.
        } break;
    }
}
```

Listing 2—Fireworks, balloons, water jets second pass

```
#define DONE    -1
#define NOP     0
#define START_BALLOON_DROP    1
#define START_FIREWORKS      2
#define START_WATER_JETS     3

#define QUERY_BALLOON_DROP    4
#define QUERY_FIREWORKS      5
#define WAIT_FOR_WATERJET_SYNC 6

int BD_Delay, FW_Delay; // Delays
extern int WindSpeed;
extern int Humidity;
extern int FireworksStartTime;
extern int BalloonDropStartTime;
extern int NowTime;

//*****[ Public ]*****
//
// The Main operating loop Second Pass at it
//
//*****
void main_pass1(void) {
    int NextCmd;
    InitHardware(); // Init all the hardware
    InitSoftware(); // Init all the software

    // Load Music Time Line
    // Wait for start signal

    // Operate the Music Time Line Interpolator with water Jet Synchronizer.
    while (NextCmd != DONE) {
        NextCmd = GetNextCmd(); // Get the next command from scheduler
        info
        switch (NextCmd) {

            case NOP: { // Do nothing
                } break;

            case QUERY_BALLOON_DROP: {
                BD_Delay = GetBD_Delay(WindSpeed); // Ask for the delay amount
                } break;

            case START_BALLOON_DROP: {
                if ((BalloonDropStartTime + BD_Delay) < NowTime) {
                    StartBalloonDrop(WindSpeed); // Send the command to
                                                    // start the balloons
                }
            } break;

            case QUERY_FIREWORKS: {
                FW_Delay = GetFW_Delay(WindSpeed, Humidity);
            } break;
            case START_FIREWORKS: {
                if ((FireworksStartTime + FW_Delay) < NowTime) {
                    StartTheFireworks(WindSpeed, Humidity);
                }
            } break;

            case WAIT_FOR_WATERJET_SYNC: {
                // Water Jets are running on their own
                // we just need to keep synchronized
            } break;
        }
    }
} // end of void main(void) {
```

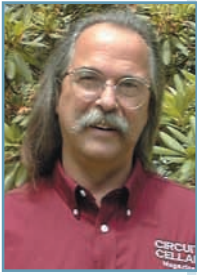
decode the WWVB time code signal (“Totally Featureless Clock (Part 3): Signal Processing,” 2010). Listing 3 on page 47 is a classic example of state machines. Using the C programming language, you will typically find state machines implemented using the switch statement. Ed’s code is an example of a good use for the C switch statement. I say that because it’s simple to see how his design and code work. As you look through code from *Circuit Cellar* and other sources, you find switch statements used frequently. Look at how straightforward some are and how obscure others have become.

In this article, I’ll revisit the topics of state machines, using them until they get overloaded and become a problem, and fixing the overload. I’ll then show you how to get ready to make the transition into a real-time operating system (RTOS).

All my code examples that use the switch statement are perhaps better described as command processors than state machines. My intent was to keep the code simple so that we could focus on the techniques. For another discussion of C code using the switch statement, refer to Simon Tatham’s 2000 document titled “Coroutines in C” (www.chiark.greenend.org.uk/~sgtatham/coroutines.html). Additional information about finite state machines is available at http://en.wikipedia.org/wiki/Finite-state_machine.

OUR WORD PROBLEM

Imagine this. Great news. Sales just called and informed us (engineering) that we won a contract for control software for the fireworks display and the closing ceremonies of the next Olympic games. (You can change this example to your latest hot project.) Here is a brief description of the assignment: there are fireworks, a music track, water fountains, and balloons released from above. And all of these must be synchronized and must operate automatically once started. Another key detail is that the music track must be well-known ahead of time and well-defined. All synchronization will coincide with the music track. The fireworks, water jets, and balloons all have different delays. For example, after the release of the restraint for



Transmit and Decode Data

Design and Implement a Keyfob Decoder

Data transmission technology has come a long way during the past few decades. Today, wireless technology is everywhere. In this article you learn how to design and implement a keyfob decoder.

In my youth, I had an appreciation for Lionel trains and Erector sets. I distinctly remember one of the projects described in the Erector set booklet incorporated electrical construction as well as mechanical. A couple of batteries were connected to the AC motor's plug, along with a couple of hand-held metal handles. In series with the batteries, I added a switch consisting of a gear turned by a crank. An insulated metal girder was positioned so that its end made and broke contact with the gear's teeth as it was rotated. This provided a repeated switch like toggling to the current running into the motor and the parallel handles.

So, I would have one of my sisters grab the handles while I cranked the little gear, providing her with an electronic "tickle" as the motor's field coils alternately energized and collapsed. To this day, I can't believe that this electrocution engine was a documented project. My three younger sisters were gluttons for the punishment I doled out. I feel like I need to apologize to them for this. Sorry, girls!

Those episodes were probably my first encounter with electricity. When I received my first Lionel train set, I became entranced with remote control. Even though this was "wired" remote control via the tracks, I could start and stop the locomotive from afar and even change its route via track switches. Today, train enthusiasts have small CCD cameras mounted in the engines that give the driver an engineer's point of view of the action. I had to use my imagination back then!

No longer do we consider "wired" to be remote control. All-out remotes are now wireless. They might be IR, RF, or even audio. Remember The Clapper sound-activated on/off switch? Many of you probably carry a remote in your pocket to authorize access to your car. When I wanted to give one of my projects an optional manual control of its X-Y positioner, I opted for a wireless link. As this project was mounted beyond my reach, I wanted to be able to manually adjust it without having to set up an extension ladder each time.

BOM

After putting together the bill of materials for a key chain RF dongle, I scratched my head and thought: "There must be an easier and cheaper way to get this done." While thumbing through a parts catalog, I came across the answer I was looking for. Linx Technologies makes a line of transmitter keyfobs. At less than \$20, it looked like the way to go. I chose a CMD-KEY5-433 keyfob, which has five push buttons and transmits an on-off keying (OOK) signal on 433 MHz. The keyfob employs an LR series transmitter with a Holtek HT640 encoder IC. While these come in 315-, 418-, and 433-MHz versions, only the 433-MHz version is precertified for FCC part 15.

PROTOCOL

For those of you unfamiliar with Holtek encoder/decoders, here's a little background. The Holtek transmission is not the simple 8-bit

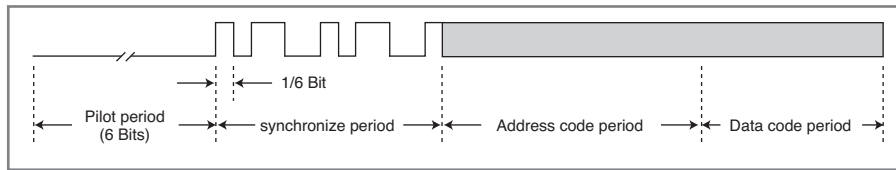


Figure 3—A transmission consists of 2 sync bits followed by 10 address bits and 8 data bits. Each transmission is separated by at least 6 bits of silence.

the internal PCB, but not necessary unless there are security issues. (You want to select one of the other available addresses.) The fob is offered with up to five push button switches. Any unused data inputs are floating. Buttons are floating when not pushed and pulled to VCC while pushed.

To use this keyfob transmitter with your circuitry, you must also use an RF receiver. The suggested RXM-433-LR receiver costs less than \$15 and requires no external tuning components. There are other, less expensive, alternatives like Microchip Technology's rfRXD0420, but they require a couple of dozen external components. I used the Linx LR in this project, but I might take a closer look at the alternatives for a future column.

DECODER IN CODE

Most of your projects will contain a microcontroller. Today, these are inexpensive and can replace other logic and timing circuitry. In many RF projects, communication consists of transferring a serial bitstream to and from UARTs over a RF link. If you were to start from scratch, you could use the common serial data approach. But security issues preclude using this approach because it's easy to duplicate. Using a nonstandard data format makes it difficult for sniffers to determine what's going on.

As you saw earlier, a UART can't decode the Holtek format. It has sync, address, and data information squashed into a single 20-bit transmission. When using a hardware encoder and decoder the format is fixed: 2-bit sync, 10-bit address, and 8-bit data. At the encoder, there really isn't any difference between the address and the data inputs. With a hardware decoder, this 20-bit transmission is treated as a 10-bit address and 8-bit data. Using a microcontroller

provides great flexibility. You can choose to emulate the hardware decoder exactly (as in this project) or change the way data is interpreted. Any of the keyfob's incarnations offer less than eight buttons. What happens to the remaining data inputs that are not used? They are not used, and thus wasted. Using a software decoder, you can treat these unused data bits as address bits. This increases the potential receiver pool. My five-button keyfob can add three additional address bits to the available 10 bits, going from 1,024 addresses to 8,196. A hardware decoder would accept multiple addresses because it uses only 8 bits for address matching. You can play around with other possibilities like swapping the address and data bits, which would really frustrate the hardware decoder. But this project's purpose is to show how to implement the hardware decoder in

software. So, I'll leave permutation to your imagination.

This project is based on the assumption that you'll do other things with the microcontroller, so the decoder support routines can't tie up the execution. To do this, you need to use two interrupts to eliminate the wait (i.e., inter-bit sampling delays.) Refer back to Figure 3. You know (and can measure) that the bit times for the transmitted format is nominally 2.6 ms per bit, which is calculated in the following fashion: $(1/76 \text{ kHz}[\text{clock speed}]) \times 33[\text{clock divisor}] \times 6[\text{cycles/bit}]$ at approximately 3 V. Each bit is made from two patterns, each of which is three cycles in length. Each pattern starts with a falling edge (and a low state). One cycle later it can change state. And one cycle after that it must end high. Two patterns make up a bit and can be one of four different types: one, zero, open, or sync.

To decode these patterns you need a peripheral to indicate a falling edge. It can be an external interrupt, a change of state interrupt, a counter/timer trigger, or something else depending on your microcontroller. I chose the Capture/Compare/PWM module (CCP1). The decoding process happens

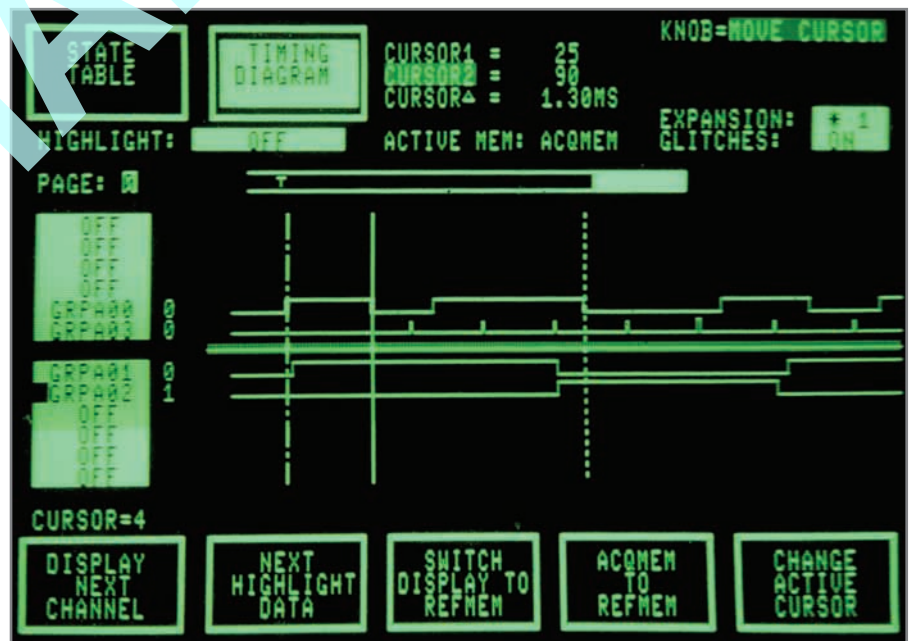


Photo 1—I used some output bits placed within my code to allow my logic analyzer to display not only the encoded transmission (the top trace), but other important events like bit sample timing in the second trace. The first vertical marker is placed at the start of the transmission with the second and third markers surrounding the first half of a bit.



Once More, With Feeling

An MCU + FPGA Without the Compromises

The idea of combining an MCU and FPGA on the same chip isn't new. Although it looks good on paper, there have been as many misses as hits. What's the difference that separates winners from losers? Now, Actel is taking a shot at it. Do they have the answer?

Looking back, I can see it was around 10 years ago when I covered the Triscend TE505 ("SoC it to Me," *Circuit Cellar* 116, 2000). It combined a fast '51 MCU with Xilinx-style (i.e., SRAM look-up table) programmable logic. Around the same time, I wrote an online column covering Atmel's FPSLIC ("Atmel Gets Huge," *CC Online*, January 2000), which married their AVR MCU and AT94-family FPGA. Perhaps you even recall when Motorola (now Freescale) made a splash with their "Core Plus" combining a Coldfire 32-bit core with fine-grained SRAM FPGA know-how acquired from UK-based Pilkington Microelectronics. Or how about QuickLogic and their QuickMIPS "ESP" (Embedded Standard Product)? Then, in "Soc Hop" (*Circuit Cellar* 128, 2001), I covered yet another variation on the theme from a then seemingly unlikely source, the Cypress Semiconductor PSoC. Clearly, the turn of the century was the golden age for integrated processors and FPGAs. So where do we stand today?

Triscend is history. It's interesting that their demise was practically more newsworthy than their short life. Back in 2004, they were doing the start-up death rattle and there were rumors ARM was going to buy them. Instead, Xilinx swooped in and took control and ... nada, the corporate equivalent of taking their ball (FPGA patents) and going home.

FPSLIC apparently still lives, at least on the Atmel website. But one has to wonder if it isn't a "zombie" product, noting for example that the "latest FAQ" (or maybe that's "inFAQ") was in 2005.

The Motorola "Core Plus" made a splash alright, except it was a belly flop. It came and went so fast you might have wondered if you'd just imagined it. And if QuickLogic really had ESP (as in extra-sensory perception), they might have foreseen announcing "the end-of-life of our QuickMIPS products due to low customer demand."^[1]

On the other hand, the Cypress PSoC is a huge hit. Just ask Cypress founder T. J. Rodgers, not one to mince words, when he says the PSoC "has obviously been a company-changing success" and "is now the flagship product at Cypress."^[2]

There are, no doubt, some lessons here. Let's see if we, and Actel, have learned them.

ARM & A LEG

Indeed, Actel's new SmartFusion isn't even the first chip to combine an ARM core with an FPGA. Triscend had announced one in their heyday, although I can't recall if it even got beyond the lab and a press release before the curtain came down.

FPGA heavyweight Altera anted up their own ARM+FPGA hybrid with Excalibur (T. Cantrell,

as a development tool. In the latter role, a ProASIC FPGA serves as a built-in “Low-cost Programmer” and debug port for the SmartFusion device. Notice as well that the 2×10 header at the top of the board provides a standard RealView programming and debug header-compatible with popular ‘M3 MCU program/debug pods, such as the Keil U-LINK and IAR J-Link. Right from the get-go, the multifaceted debug and programming scheme establishes a theme that will be repeated: SmartFusion may be one chip, but from the designer’s perspective, it largely retains

the look and feel of a two-chip setup (i.e., an ‘M3 MCU and an FPGA).

Along the bottom of the board is a pin header that provides access to 100 pins worth of the SmartFusion I/O. I understand small (1/20” centers) is beautiful, but I wonder if it wouldn’t have been better to stick with a standard 1/10” header in the interest of compatibility with the jumpers and breadboards everyone already has at hand. Fortunately, the board has a sufficient complement of the usual I/O (e.g., push buttons, LEDs, a potentiometer, and a temperature diode), so

you can get started without having to deal with the connector issue.

One unique embellishment is the little OLED display on the upper right, which serves as a mini-me demo/debug console, filling a role historically served by a clunky LCD. Yes, the display is tiny, but there’s enough resolution (96×16) and brightness that characters (e.g., 12 characters \times 2 lines with a 5×7 font) are surprisingly clear and it’s quite usable in practice. Very cute and easy to interface (I-C), so I bet you’ll see more of these little nuppies popping

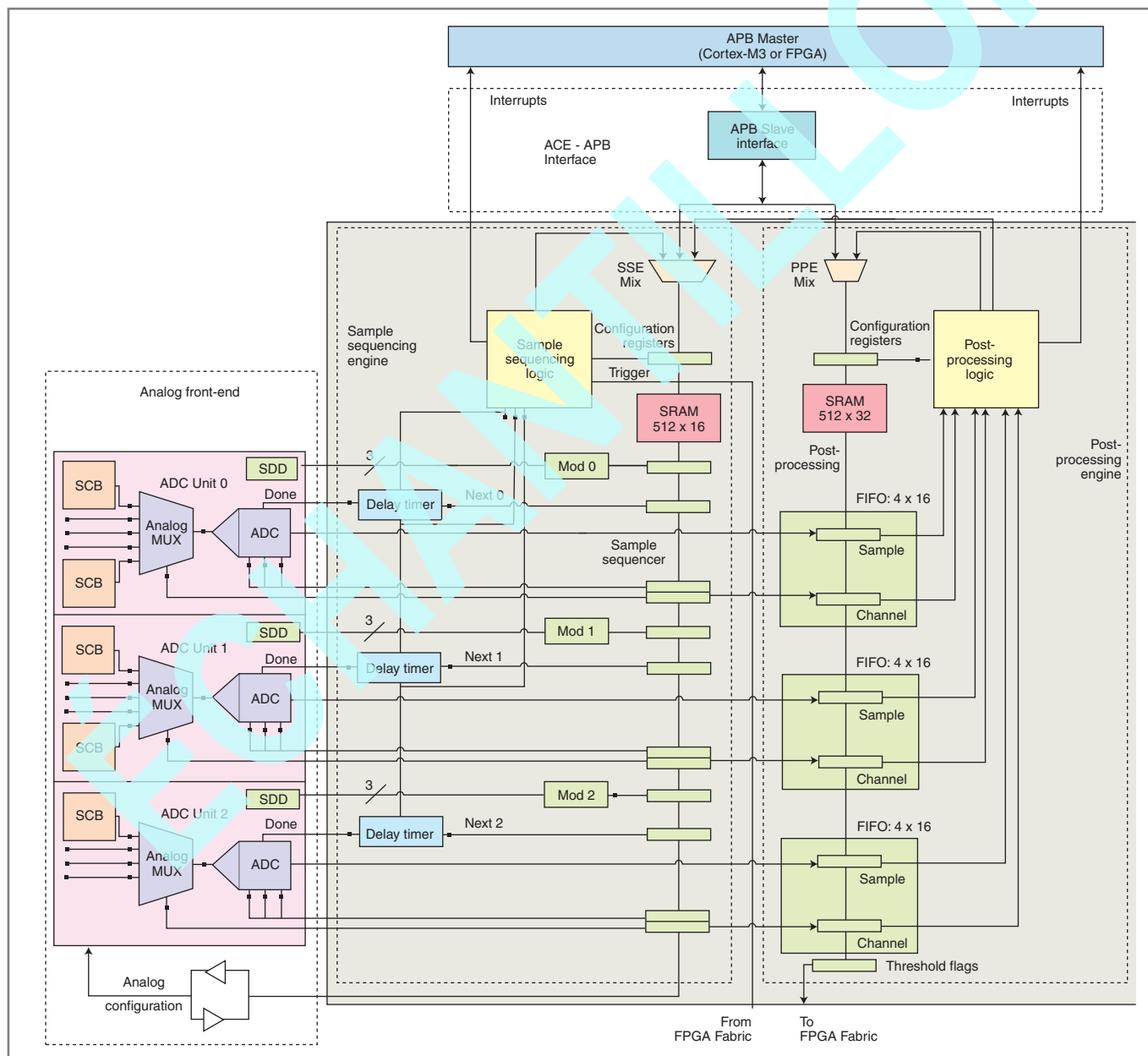
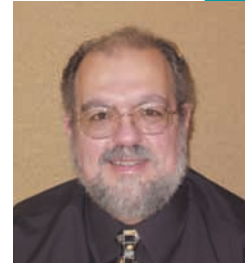


Figure 2—One key advantage of SmartFusion over an admittedly lower-cost two-chip solution (i.e., MCU + FPGA) is significant analog capability in the form of an “Analog Compute Engine” (ACE). By handling key signal-conditioning and processing tasks, the ACE meets strict timing requirements while freeing the ‘M3 for higher-level application.

PRIORITY INTERRUPT



by Steve Ciarcia, Founder and Editorial Director

Mobile Metamorphosis

I consider myself to be a technically sophisticated Luddite. Yes, I realize that is probably an oxymoron, but it might be the best way to describe a guy who has the technical knowledge to design gadgets that enhance a contemporary world but who then prefers not to use them himself. It's not as bad as wanting to return to rotary-dial phones mind you, but neither do I want to join a new generation that seems to get off on streaming live video and tweets while arguing with a meter maid giving them a parking ticket.

Certainly, if we have to pick a significant invention that has sparked our present cultural metamorphosis, it's the advent of the cell phone and wireless Internet communications. In the last 30 years, we have gone from simple newspapers, land-line telephones, and terrestrial TV with selective intermittent interaction, to complete and constant physical, behavioral, and emotional connection to virtually the entire world and everything in it.

This isn't a rant. It's an observation. I had my first cell phone in 1981 (a Diamondtel MESA-55 that was \$5,200 in 2010 dollars). I've owned every advance in mobile communications from the day the first cell tower was turned on—right up until everyone jumped on smart phones! It's a long story, but the Luddite in me just couldn't put a handle on the touchy-feely cultural phenomenon. My normal cell phone (albeit a phone costing \$500) operating mode was that I disabled voicemail, never used Bluetooth, never sent a text message, never took a picture, and always shut it off except when travelling.

Certainly, I'm no poster boy for mobile communications, but I did recognize the value of having an Internet connection away from home. My understanding, or misunderstanding in retrospect, was that using a small laptop computer with an Internet connection was the best way to go. About two years ago, I bought an Acer netbook with XP and installed a Sprint broadband card. For \$70 per month, I became the latest-and-greatest mobile Internet hotspot. ;-) After some experience, however, I found that a netbook and broadband card combination has to be one of the dumbest "compact" mobile computing platforms on the planet!

I know there is some jeopardy in admitting to you that I made a mistake, but let's chalk it up to cultural ignorance. When smart phones were introduced, especially the iPhone, all I saw was a crazed crowd of users who would simply be doing a lot more of the same mobile stuff that I already didn't do. Furthermore, the iPhone was only sold to function on a network I didn't care for, Adobe "flash" was considered to be a swear word, and it only used software sourced and approved by a company very closely resembling a communist government.

It wasn't until a couple months ago when my close friend, George Splane, bought an iPhone and started showing me some of the things it did that I saw the light. Yes, it could tweet, take pictures, etc.—but, more importantly, it did an amazing number of other chores with an efficiency that I hadn't realized. Certainly, I could accomplish similar tasks on my netbook, but they probably wouldn't be free applications, and it would never be as easily mobile.

Ultimately, my procrastination lasted until two days ago when I bought a Droid X. I was astonished at the powerful technology in smart phones and all that I had been missing. This description of my metamorphosis is not meant to be read as a comparison between phone brands or services (Droid X uses Verizon). I chose the Droid X because size wasn't an issue, and I wanted as much processing power and screen resolution as available. More importantly, with all my monitoring systems and webcams, I WANT FLASH!

I will be the first to admit that I made a mistake by ignoring an important technology for so long. In fact, I think that "smart phones," or whatever the pad/tablet/phone device configuration, are the best thing since sliced bread. As I write this, I've only had the Droid X for two days, but I've already made videos, sent e-mails, programmed in 50 favorite websites, and downloaded a couple dozen free applications like weather maps, bar code scanning, Wi-Fi scanning, news channels, wine selection, music radio, and cocktail recipes. It's really neat.

The lesson to learn here isn't so much that I've finally overcome my ignorance, but that now that I know the advantages of smart mobile communication, I can never go back. This is a wise lesson for embedded control designers as well. I may be all too typical of the stogy executives who decide what to market and what to buy. The savvy staff engineer who keeps a hook in every design so it is compatible or can be enhanced by smart mobile communication will get the praise and reward when management finally catches up. Remember, even Luddites see the light eventually. When it happens, these people will switch loyalties to the latest and greatest in a flash (pun intended). You want to be the one providing the light and getting the credit.

steve.ciarcia@circuitcellar.com

A handwritten signature in black ink, appearing to read "Steve".