



circuit cellar

REAL-TIME VEHICLE TRACKER

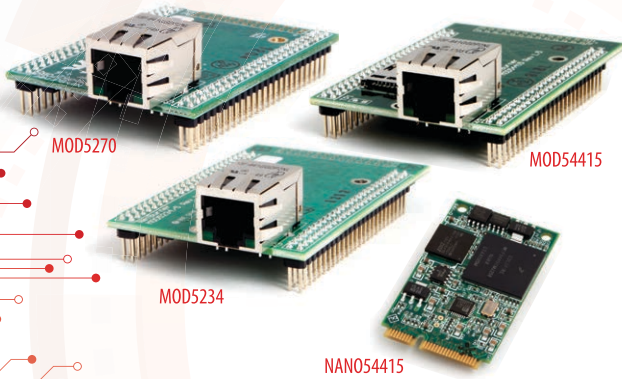


Build a system that works with a mobile app, a GPS, and a smartphone interface



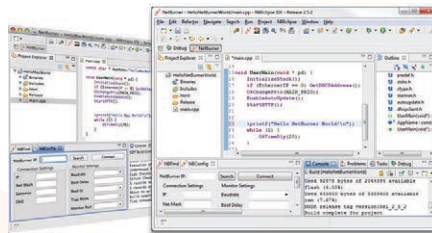
- Top EE Tips | Q&A: Embedded Systems Consultant and Hacker
- Automated Vehicle Locator | Bit Banging Explained | GPS-Based Sun Tracker
- Data Encoding | FPGA Reconfiguration | Customizing the Linux Kernel
- Voltage Step-Up Tools | RFID Circuits ■ Bluetooth Low Energy's Impact

Ethernet Core Modules with High-Performance Connectivity Options



- **MOD5270**
147.5 MHz processor with 512KB Flash & 8MB RAM · 47 GPIO
3 UARTs · I²C · SPI
- **MOD5234**
147.5 MHz processor with 2MB flash & 8MB RAM · 49 GPIO · 3 UARTs
I²C · SPI · CAN · eTPU (for I/O handling, serial communications,
motor/timing/engine control applications)
- **MOD54415**
250 MHz processor with 32MB flash & 64MB RAM · 42 GPIO · 8 UARTs
5 I²C · 3 SPI · 2 CAN · SSI · 8 ADC · 2 DAC · 8 PWM · 1-Wire[®] interface
- **NANO54415**
250 MHz processor with 8MB flash & 64MB RAM · 30 GPIO · 8 UARTs
4 I²C · 3 SPI · 2 CAN · SSI · 6 ADC · 2 DAC · 8 PWM · 1-Wire[®] interface

Add Ethernet connectivity to an existing product, or use it as your product's core processor



The goal: Control, configure, or monitor a device using Ethernet

The method: Create and deploy applications from your Mac or Windows PC. Get hands-on familiarity with the NetBurner platform by studying, building, and modifying source code examples.

The result: Access device from the Internet or a local area network (LAN)

The NetBurner Ethernet Core Module is a device containing everything needed for design engineers to add network control and to monitor a company's communications assets. For a very low price point, this module solves the problem of network-enabling devices with 10/100 Ethernet, including those requiring digital, analog and serial control.

MOD5270-100IR.....\$69 (qty. 100)	NNDK-MOD5270LC-KIT.....\$99
MOD5234-100IR.....\$99 (qty. 100)	NNDK-MOD5234LC-KIT.....\$249
MOD54415-100IR.....\$89 (qty. 100)	NNDK-MOD54415LC-KIT.....\$129
NANO54415-200IR....\$69 (qty. 100)	NNDK-NANO54415-KIT.....\$99

NetBurner Development Kits are available to customize any aspect of operation including web pages, data filtering, or custom network applications. The kits include all the hardware and software you need to build your embedded application.

➤ **For additional information please visit**
<http://www.netburner.com/kits>



TFT Displays

Available in 1.8"-7.0" sizes

- Capacitive and Resistive touch options
- MVA, IPS, and TN technologies
- WVGA, QVGA, and VGA resolutions
- Standard stocked displays
- Development kits available
- Competitive pricing with low MOQs
- Custom options available
- Superior engineering support



NEWHAVEN DISPLAY INTERNATIONAL

For more information visit: circuitcellar.newhavendisplay.com

Contact us at: 847-844-8795



RoHS
Compliant

Issue 287 June 2014 | ISSN 1528-0608

CIRCUIT CELLAR® (ISSN 1528-0608) is published monthly by:

Circuit Cellar, Inc.
111 Founders Plaza, Suite 300
East Hartford, CT 06108

Periodical rates paid at East Hartford, CT, and additional offices.

One-year (12 issues) subscription rate US and possessions \$50, Canada \$65, Foreign/ ROW \$75. All subscription orders payable in US funds only via Visa, MasterCard, international postal money order, or check drawn on US bank.

SUBSCRIPTIONS

Circuit Cellar, P.O. Box 462256, Escondido, CA 92046

E-mail: circuitcellar@pcspublink.com

Phone: 800.269.6301

Internet: circuitcellar.com

Address Changes/Problems: circuitcellar@pcspublink.com

Postmaster: Send address changes to
Circuit Cellar, P.O. Box 462256, Escondido, CA 92046

ADVERTISING

Strategic Media Marketing, Inc.
2 Main Street, Gloucester, MA 01930 USA

Phone: 978.281.7708

Fax: 978.281.7706

E-mail: circuitcellar@smmarketing.us
Advertising rates and terms available on request.

New Products:

New Products, Circuit Cellar, 111 Founders Plaza, Suite 300
East Hartford, CT 06108, E-mail: newproducts@circuitcellar.com

HEAD OFFICE

Circuit Cellar, Inc. 111 Founders Plaza, Suite 300
East Hartford, CT 06108
Phone: 860.289.0800

COVER PHOTOGRAPHY

Chris Rakoczy, www.rakoczyphoto.com

COPYRIGHT NOTICE

Entire contents copyright © 2014 by Circuit Cellar, Inc. 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.

DISCLAIMER

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 infringing liability for constructing or operating such devices.

© Circuit Cellar 2014 Printed in the United States

VEHICLE TRACKING, BIT BANGING, AND MORE

When the campus at Penn State Erie, The Behrend College, had a growth spurt, the local transit authority provided a shuttle bus to help students who were rushing from class to class. But ridership was low because of the bus's unpredictable schedule.

So a college engineering team constructed a mobile application to track the bus (p. 20). That system inspired the cover of our June issue and complements its communications theme.

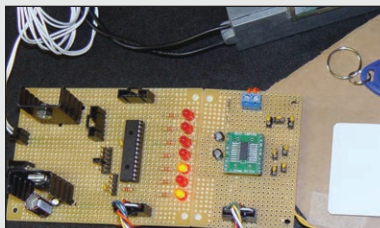
The three-part system consists of a user's smartphone running a HTML5-compatible browser, a base station consisting of an XTend 900-MHz radio connected to a Raspberry Pi single-board computer, and a mobile tracker including a GPS receiver, a Microchip Technology PIC18F26K22 microcontroller, and an XTend module.

The Raspberry Pi runs a web server to handle requests from a user's smartphone. The user then receives accurate bus arrival times.

Also aligning with this month's theme, we present an article about implementing serial data transmission through bit banging (p. 30). You'll gain a better understanding of how serial data is transmitted and received by a microprocessor's hardware UART peripheral. You'll also learn how bit banging can achieve serial communication in software, which is essential when your embedded system's microprocessor lacks a built-in UART.

Recognizing a rapidly unfolding communications trend, this issue includes an inventor's essay about how the presence of Bluetooth Low Energy (BLE) in the latest mobile devices is sparking a big boom in innovative hardware/sensor add-ons that use your smartphone or tablet as an interface (p. 80). Other communications-related articles include Part 2 of a close look at radio-frequency identification (RFID). This month's installment describes the front-end analog circuitry for the RFID base station of a secure door-entry project (p. 68).

In addition, we offer articles about adjusting your FPGA design while it's operating (p. 52), modifying the Linux kernel to suit your hardware and software designs (p. 58), tools and techniques to boost your power supply (p. 62), digital data encoding in wireless systems (p. 46), GPS orientation of a solar panel (p. 38), and an interview with Quinn Dunki, an embedded applications consultant and hacker (p. 8).



Mary Wilson

mwilson@circuitcellar.com

THE TEAM

EDITOR-IN-CHIEF

C. J. Abate

MANAGING EDITOR

Mary Wilson

ASSOCIATE EDITOR

Nan Price

ART DIRECTOR

KC Prescott

ADVERTISING COORDINATOR

Kim Hopkins

PRESIDENT

Hugo Van haecke

PUBLISHER

Don Akkermans

ASSOCIATE PUBLISHER

Shannon Barraclough

COLUMNISTS

Jeff Bachiochi, Ayse K.

Coskun, Bob Japenga, Robert
Lacoste, Ed Nisley, George
Novacek, Colin O'Flynn

FOUNDER

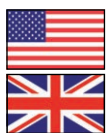
Steve Ciarcia

PROJECT EDITORS

Chris Coulston,
Ken Davidson,
David Tweed

CUSTOMER SERVICE

Debbie Lavoie

**US/UK**

Don Akkermans
+31 46 4389444
d.akkermans@elektor.com

**ELEKTOR LABS**

Wisse Hettinga
+31 46 4389428
w.hettinga@elektor.com

**GERMANY**

Ferdinand te Walvaart
+49 241 88 909-17
f.tewalvaart@elektor.de

**FRANCE**

Denis Meyer
+31 46 4389435
d.meyer@elektor.fr

**NETHERLANDS**

Harry Baggen
+31 46 4389429
h.baggen@elektor.nl

**SPAIN**

Eduardo Corral
+34 91 101 93 95
e.corral@elektor.es

**ITALY**

Maurizio del Corso
+39 2.66504755
m.delcorso@inware.it

**SWEDEN**

Wisse Hettinga
+31 46 4389428
w.hettinga@elektor.com

**BRAZIL**

João Martins
+31 46 4389444
j.martins@elektor.com

**PORTUGAL**

João Martins
+31 46 4389444
j.martins@elektor.com

**INDIA**

Sunil D. Malekar
+91 9833168815
ts@elektor.in

**RUSSIA**

Nataliya Melnikova
+7 965 395 33 36
elektor.russia@gmail.com

**TURKEY**

Zeynep Köksal
+90 532 277 48 26
zkoksal@beti.com.tr

**SOUTH AFRICA**

Johan Dijk
+31 6 1589 4245
j.dijk@elektor.com

**CHINA**

Cees Baay
+86 21 6445 2811
ceesbaay@gmail.com

OUR NETWORK

VOICE COIL

**SUPPORTING COMPANIES**

Accutrace	7	Jeffrey Kerr, LLC	78
All Electronics Corp.	78	MaxBotix, Inc.	79
ARM	29	Measurement Computing Co.	41
Beta LAYOUT, Ltd.	65	microEngineering Labs, Inc.	79
CadSoft Computer GmbH	23	MyRO Electronic Control Devices, Inc.	78
Cleverscope	19	NetBurner, Inc.	C2
Custom Computer Services	78	Newhaven Display International	1
Elektor	45	Pico Technology, Ltd.	C3
Elektor	50, 51	R.E. Smith, Inc.	61
Elprotronic, Inc.	65	Reach Technology, Inc.	78
EMAC, Inc.	19	RS Components	34-36
ExpressPCB	49	Saelig Co., Inc.	61
GHI Electronics, LLC	33	Sensors Expo & Conference 2014	25
HuMANDATA, Ltd.	78	Technologic Systems	9, 11
IAR Systems	15	Techsol Engineering, Inc.	79
Imagineering, Inc.	C4	Triangle Research International, Inc.	79
Ironwood Electronics	78		

NOT A SUPPORTING COMPANY YET?

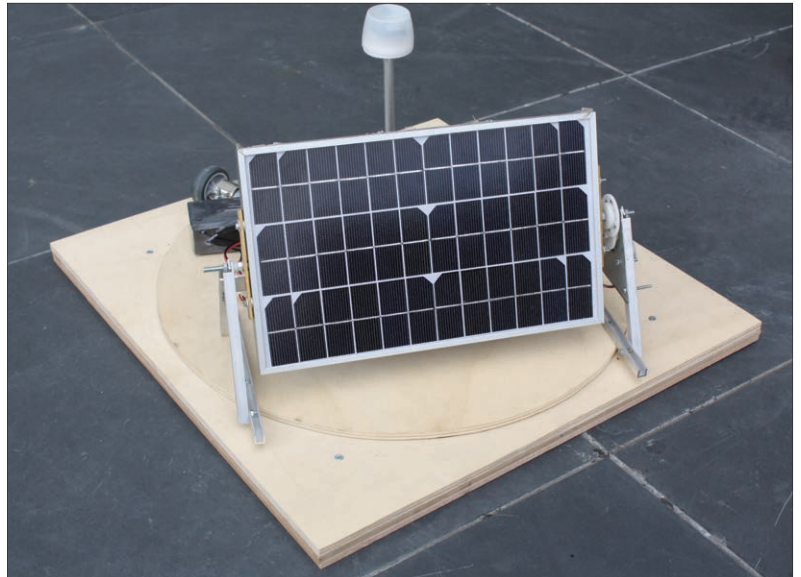
Contact Peter Wostrel (circuitcellar@smmarketing.us, Phone 978.281.7708, Fax 978.281.7706) to reserve your own space for the next edition of our members' magazine.

CONTENTS



JUNE 2014 • ISSUE 287

COMMUNICATIONS



SUN-TRACKING SYSTEM

CC COMMUNITY

06 : CC WORLD

08 : QUESTIONS & ANSWERS

Embedded Consulting, Software Developing, and Hacking

By Nan Price

Quinn Dunki on her computer game company, hackerspaces, pinball-machine project, and more

INDUSTRY & ENTERPRISE

13 : PRODUCT NEWS

19 : CLIENT PROFILE

Measurement Computing Corp.
(Norton, MA)

FEATURES

20 : Build an Automated Vehicle Locator

By Chris Coulston, Daniel Hankewycz, and Austin Kelleher

Pinpoint a moving vehicle via a smartphone web interface to a GPS mobile tracker and base station

30 : Bit Banging

By Shlomo Engelberg

Achieve serial communication without a built-in UART

38 : The Sun Chaser

GPS Reference Station Design

By Sjoerd Brandsma

Use GPS, an external compass, and a stepper motor to continuously orient a solar panel toward the sun

COLUMNS

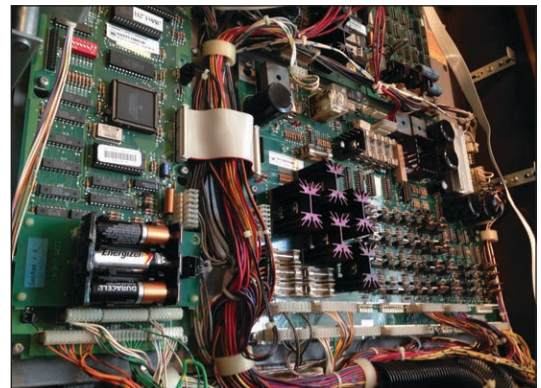
46 : THE CONSUMMATE ENGINEER

Wireless Data Links (Part 4)

Data Encoding

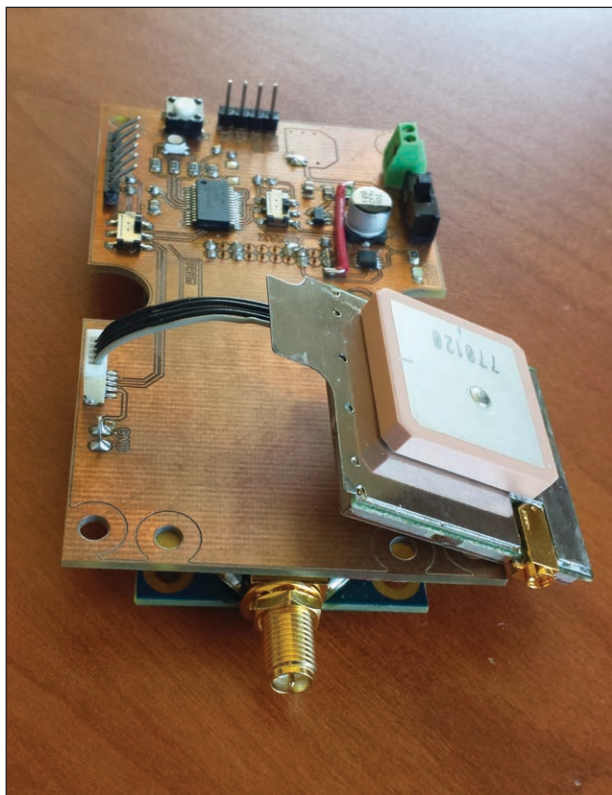
By George Novacek

An overview of digital data encoding methods and solutions for multipath distortion

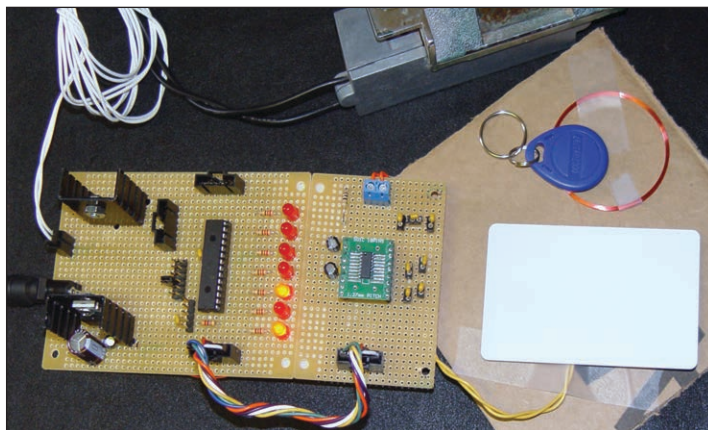


PINBALL MACHINE HACK

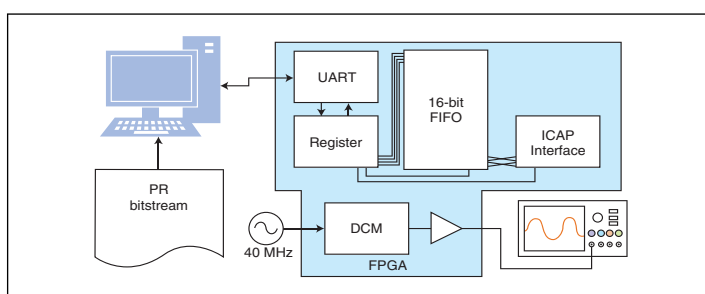
CONTENTS



MOBILE VEHICLE TRACKER



FRONT-END ANALOG CIRCUITRY FOR AN RFID BASE STATION



FPGA RECONFIGURATION DURING OPERATION

52 : PROGRAMMABLE LOGIC IN PRACTICE Partial FPGA Configuration

By Colin O'Flynn

Adjust your FPGA design during operation

58 : EMBEDDED IN THIN SLICES Linux System Configuration (Part 1)

The Linux Kernel

By Bob Japenga

Tweak the Linux kernel to suit your design's hardware, software, and debugging needs

62 : THE DARKER SIDE Voltage Step-Up Techniques

By Robert Lacoste

Integrate transformers, voltage multipliers, boost converters, and charge pumps to increase your power supply

68 : FROM THE BENCH Passive RFID Tagging (Part 2)

Front-End Analog Circuits

By Jeff Bachiochi

Build a secure door-lock entry system with an analog RFID front-end base station

TESTS & CHALLENGES

74 : CROSSWORD

75 : TEST YOUR EQ

TECH THE FUTURE

80 : Bluetooth Low Energy Changes the "Wireless Landscape"

By Eric VanWyk

BLE 4.0 in smartphones and tablets fuels growth and innovation in wireless hardware/sensor add-on devices



WIRELESS MULTIMETER ON A SMARTPHONE

CC WORLD



TOP ELECTRICAL ENGINEERING TIPS

By CC & EIM Staff (US)



Visit circuitcellar.com regularly for new EE Tips.

The demand for electrical engineering tips and tricks has been steadily increasing among engineers in *Circuit Cellar's* international community since January 2014. In addition to logging increased pageviews, the "EE Tips" posts on circuitcellar.com are being shared throughout the community via social media and e-mail.

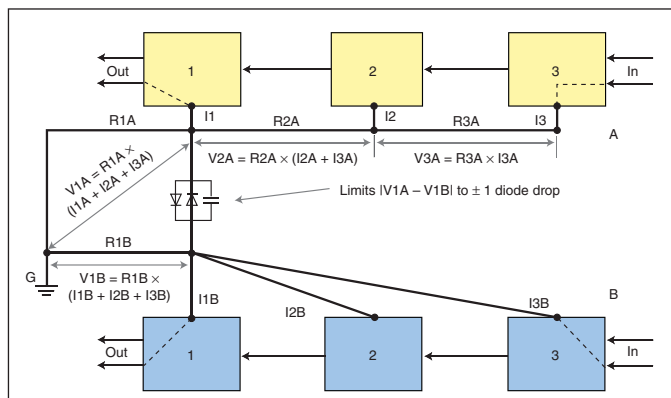
As of this issue's printing, the following three tips are the most widely read.

Triangulation, Trilateration, or Multilateration?

EE Tip #125

By Clemens Valens

Local Positioning System (LPS) and GPS (not just the US system) both use several transmitters to enable a receiver to calculate its geographical position. Several techniques are possible, each with its advantages and drawbacks. The important thing about all these techniques is the notion of a direct path (line of sight, or LoS). In effect, if the transmitter signal has not taken the shortest path to the receiver, the distance between them calculated by the receiver will be incorrect, since the receiver does not know the route taken by the radio signal. Three mathematical techniques are usually used for calculating the position of a receiver from signals received from several transmitters: triangulation, trilateration, and multilateration. The last two are very similar, but should not be confused. Read more: <http://bit.ly/QUNEZr>



Both sections, A and B, may be on the same PCB with separate ground planes (e.g., analog and digital). The diodes and the capacitor between the planes limit potential differences due to ground bounce and so forth. Broken lines inside boxes 1 and 3 indicate ground-referenced, non-symmetrical inputs and outputs.

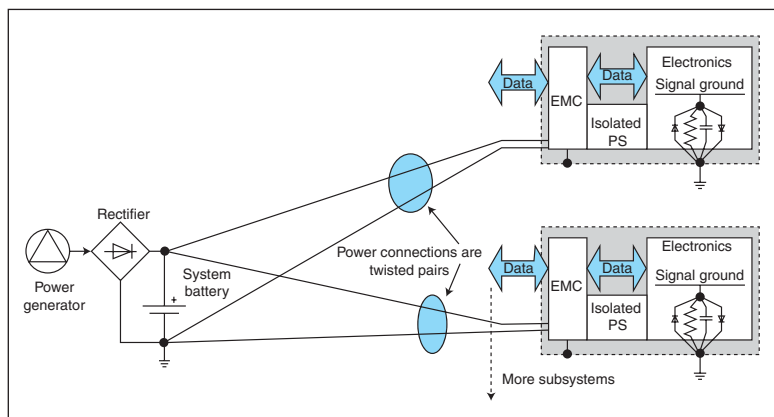
A Fundamental Rule of Grounding

EE Tip #124

By George Novacek

Quantum mechanics notwithstanding, our world is analog. And so despite our fascination with everything digital, we need interfaces to provide bridges for our analog reality to cross over to the digital paradigm and then back again. Is there a common denominator that binds these two worlds together regardless of their many conceptual differences? In my mind, it is grounding.

Figure 1 shows the fundamental rule for grounding. By "ground" I mean the common 0-V potential to which signals are referenced. The "chassis ground," if grounding conductors had 0- Ω impedance, would also be 0 V—but, unfortunately, it never is. Yet there are still systems that are sufficiently insensitive to ground potential differences. They use the chassis for the signal and power returns. At one time, this was the way cars had been wired. Read more: <http://bit.ly/1flrw6e>



This shows the connecting of subsystems to avoid ground loops.

Op-Amp vs Comparator

EE Tip #128

By Michael Holzl

Practically every lecture course or textbook on electronics describes how to use an operational amplifier as a comparator. Here we look at the possibility in more detail and see how it can often be a very poor idea. The idea behind the comparator configuration is simple. An op-amp has a very high open-loop DC gain, which means that even a tiny differential input voltage will drive the output to one extreme or the other. If the voltage at the non-inverting ("+") input is greater than that at the inverting ("−") input, the output goes high; otherwise, the output goes low. In other words the two voltages are compared and the output is a binary indication of which of the two is the greater. Read more: <http://bit.ly/1j8pNQP>

PRINTED CIRCUIT BOARDS

THINK YOU CAN FIND PCB PRICES THAT BEAT OURS?

WE DARE YOU.



OUR GUARANTEE:

We are so confident in our PCB pricing, we dare you to find lower prices! If you do, we will match the price AND give you \$100 towards your next order!

Visit us at www.PCB4u.com and see why our pricing can not be beaten!

- From same day quick turn prototype to production in under 10 days
- Full CAD and CAM review plus design rule check on ALL Gerber files
- Materials: Fr4, Rigid, Flex, Metal Core (Aluminum), Polyimide, Rogers, Isola, etc.
- HDI Capabilities: Blind/Buried Microvias, 10+N+10, Via-in-Pad Technology, Sequential Lamination, Any Layer, etc.
- Our HDI Advantage: Direct Laser Drilling, Plasma De-Smear Technology, Laser Micro via Conductive Plate Shut, etc.

Accutrace[®] inc.

www.PCB4u.com • sales@PCB4u.com • (408) 748-9600



QUESTIONS & ANSWERS

Embedded Consulting, Software Developing, and Hacking

An Interview with Quinn Dunki

Quinn Dunki is more than just a hacker. This Los Angeles, CA-based embedded applications consultant and software game developer enjoys working on her homebrew 8-bit computer and dreams of a future filled with hackerspace-type libraries.—Nan Price, Associate Editor



NAN: Tell us about your computer game company, *One Girl, One Laptop Productions* (<http://quinndunki.com>). How did the company begin?

QUINN: I had been in the AAA games industry for most of my career. I've been making games in my spare time since I was six years old, but the "actually-getting-paid-for-it" time started in the 1990s with the Nintendo 64.

I've written games on everything from the

Apple II to the Playstation 3. I worked at various companies including Bungie Studios and 3DO.

My longest stint was eight good years at a small studio called Pandemic in Los Angeles, CA. In 2009, the company was in financial trouble and was sold to Electronic Arts with the intention it would keep it going. Electronic Arts opted to close the studio down shortly thereafter. We got some severance with our walking papers, and I decided to spin that money into *One Girl, One Laptop Productions*.

This was just at the tail end of the initial gold rush on Apple's iOS platform, and it still seemed like there was money to be made there. Unfortunately, there was subsequently a mad rush to the bottom on pricing for iPhone games. Before I could establish a presence, the space got very crowded almost overnight. Low-volume, high-quality indie games became financially unviable (though I think they're coming back now). I still do independent game development on the side, but my primary business now is consulting and hired-gun engineering for other companies needing mobile or embedded applications.

NAN: Describe some of the software *One Girl, One Laptop Productions* develops. Do you have a favorite?

QUINN: As much as I love games, my true love is engineering itself. My favorite projects always end up being the ones with the most complex challenges. I don't think I could pick just one.

A good recent example is the *Oloclip*, which is a combination photography app and lens attachment for the iPhone. The main killer feature is real-time barrel distortion correction that made for some very interesting development challenges.

Quinn has two workspaces: **a**—She uses one for the "small clean stuff." **b**—The other is used for the "big dirty stuff."



SUPERIOR **EMBEDDED** SOLUTIONS



DESIGN YOUR SOLUTION TODAY
CALL 480-837-5200

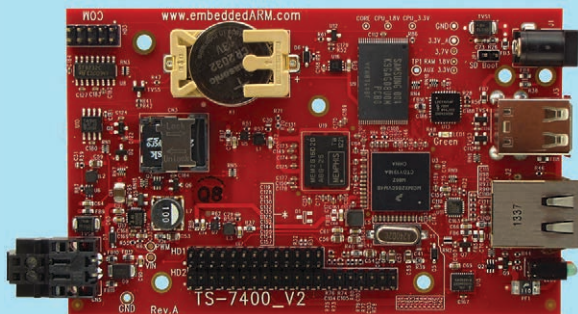
www.embeddedARM.com

NEW!

Single Board Computers

TS-7400-V2 Low Power & Low Cost

Backwards Compatible with TS-7400



pricing starts at
\$119
qty 1
\$84
qty 100

Features:

- 450MHz ARM CPU
- Up to 256MB RAM
- 2GB NAND Flash
- 2x SD Card Socket
- 1x 10/100 Ethernet
- 2x USB Host
- 39x DIO, 4x ADC
- 4x Serial ports
- 1x BBRTC, 1x CAN
- 1x Temp Sensor

Benefits:

- 2x faster, more features, reduced cost
- Low power with 120mW standby mode
- -40 to +85C, 100% soldered-on components
- Easy development w/ Debian and Linux 2.6
- Boots quickly to your Embedded Application
- Guaranteed available until 2025

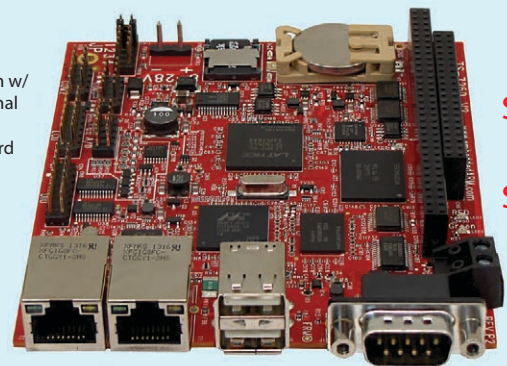
NEW!

Single Board Computers

TS-7250-V2 High Performance

Backwards Compatible with TS-72xx

shown w/
optional
micro
SD Card



pricing starts at
\$199
qty 1
\$165
qty 100

Features:

- Up to 1GHz ARM CPU
- 512MB RAM
- 4GB eMMC Flash
- 2x SD Card Socket
- 2x 10/100 Ethernet
- 2x USB Host
- 1x USB Device
- 6x Serial ports
- 75x DIO, 1x CAN
- 1x PC/104 Connector

Benefits:

- Hardware Flexibility with On-board FPGA
- Several control I/O interfaces
- Launches your application in under a second
- Easy development w/ Debian and Linux 2.6
- High Data Reliability with DoubleStore
- -40 to +85C Industrial temperature range



We've never discontinued a product in 30 years



Embedded systems that are built to endure



Support every step of the way with open source vision



Unique embedded solutions add value for our customers

QUESTIONS & ANSWERS

“Hacking is more than a hobby—it’s kind of a way of life. It’s part creativity, part environmentalism, part self-reliance, and all good times.”

Much like game consoles, working on mobile devices is often about taking a well-understood algorithm and making it work on a platform so small that nobody thinks it will be possible. On AAA games, I used to try and build complex artificial intelligence (AI) systems that ran in 3 ms of frame time. Now I’m trying to cram gigabyte-scale image processing systems into devices with little memory, minimal graphics processing units (GPUs), and slow CPUs. They are similar challenges with completely different contexts. Some days it feels like you’re trying to model high-energy particle physics on a washing machine, but it’s a great when you finally do solve a problem like that. It’s the engineering of the thing that’s exciting, whatever that thing is this week.

Another favorite has been the ICEdot project. It’s a health and safety sensor system that works with your mobile device and is targeted at athletes and coaches. It’s a fun mix of mobile and embedded systems development and it has pushed my skill set into a number of new areas—in particular, Bluetooth Low Energy (BLE), which is an exciting new technology. ICEdot is on the bleeding edge of that, and it’s been a big challenge to use it in the real world.

NAN: What types of projects did you work on while you were a Senior Engineer at Pandemic Studios?

QUINN: I started my tenure there on a squad tactics training simulator Pandemic was building for Simulation, Training, and Instrumentation Command (STRICOM), an experimental technology branch of the US Army. It’s a long story, but that simulator was later spun into a series of Xbox games called *Full Spectrum Warrior*.



To prolong her toothbrush’s life, Quinn replaced a toothbrush battery with a nickel-cadmium battery and added wires to the old battery’s PCB mount points.

The biggest project I worked on was an open-world game set in World War II called *Saboteur*. Unlike the usual shooter format the WWII genre is littered with, this was a third-person action-adventure game with a noir art style. *Saboteur* was a hugely ambitious project, and the awesome team there solved some very big challenges. We did things with physics, rendering, AI, clambering, animation, toolchains, content streaming, and game design that no game had done before. As so often happens with AAA games, the marketing budget was pulled at the last moment, so you can add it to the long list of “Greatest Games That Nobody Played.”

NAN: Your blog-style website BlondiHacks (<http://quinndunki.com/blondihacks>) features hacking projects involving everything from development boards to two-layer PCB etching. Tell us about the types of projects you enjoy hacking.

QUINN: BlondiHacks is my outlet for whatever whim that comes to mind as far as hacking. I think hacking is more than a hobby—it’s kind of a way of life. It’s about shaping your environment to be what you think it should be. It’s about saving things from landfills and giving new life to forgotten or underappreciated artifacts. It’s part creativity, part environmentalism, part self-reliance, and all good times.

It’s fun to talk about stuff you’re doing, but most of my flights of fancy are so obscure or odd that only a select few would find them interesting. The power of the Internet is that it connects all of us oddballs to each other. Hence, BlondiHacks.

NAN: How did you become interested in technology?

QUINN: I don’t recall a time when I wasn’t interested, honestly, so that transition must have occurred before my brain was retaining memories. What age is that? Three? Four?

I may have been born with a multimeter in my hand (though my mom would probably have noted that in the medical report). My mom likes to say, the day they brought the Apple II into the house (when I was around age five) I crawled up on the stool and haven’t moved since.

NAN: What was your first project?

QUINN: That’s difficult to say, since my life is a series of endless overlapping projects. As soon as I was old enough to hold a soldering iron, I built a lot of things from the seminal Forrest Mims book RadioShack sold. You know

SUPERIOR **EMBEDDED** SOLUTIONS

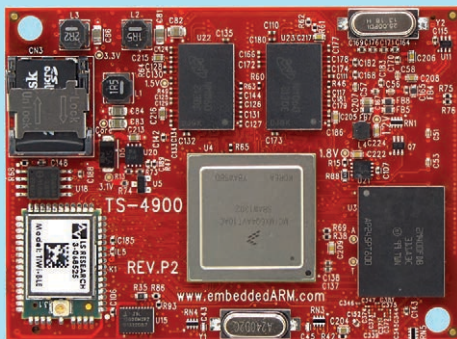


DESIGN YOUR SOLUTION TODAY
CALL 480-837-5200

www.embeddedARM.com

Computer-on-Modules

State of the Art Embedded Design



pricing starts at
\$139
qty 1
\$99
qty 100

- Up to 1.2GHz ARM w/ 2GB RAM
- Fanless, low power designs
- All parts soldered on, no moving parts
- DoubleStore SD for reliable storage
- User-programmable FPGAs
- -40 to +85C Industrial temperature range
- Easy development w/ Android, Debian and Linux
- Sub-second boot to Linux
- COTS carrier boards available or design a custom solution with reduced design time and complexity

Computer-on-Modules include:

NEW!

- TS-4200: Atmel ARM9 w/ super low power
- TS-4600: 450MHz low cost w/ 2 Ethernets
- TS-4710: Up to 1066MHz PXA168 w/ video
- TS-4712: like TS-4710 + 2 Ethernets
- TS-4720: like TS-4712 + 4GB eMMC Flash
- TS-4800: 800MHz FreeScale iMX515 w/ video
- TS-4900: 1.2GHz QuadCore i.MX6 w/ WiFi

NEW!

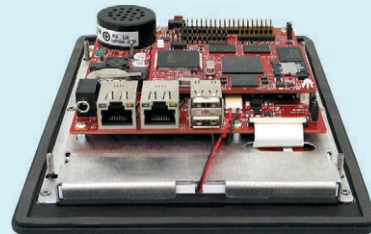
NEW!

Touch Panel Computers

Panel Mount or Fully Enclosed



series starts at
\$369
qty 100
\$409
qty 1



Features can include:

- 5-inch, 7-inch and 10-inch touchscreens
- Fanless operation from -20 to +70C
- Up to 1.2Ghz ARM CPU
- Up to 2GB RAM, 4GB eMMC Flash
- 2x microSD with DoubleStore
- 2x Ethernet, 2x USB Host
- CAN, RS-232, SPI, I2C, DIO
- 1x RS485 2W-Modbus
- Optional cellular, WIFI & XBEE radios
- Support Android and Linux w/ sub-second boot
- Headphone connector & speaker



We've never discontinued a product in 30 years



Embedded systems that are built to endure



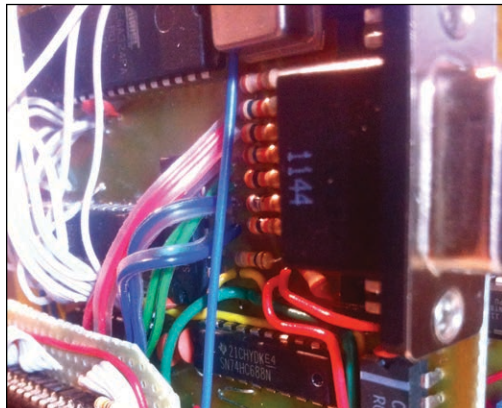
Support every step of the way with open source vision



Unique embedded solutions add value for our customers

QUESTIONS & ANSWERS

Quinn's homebrew computer, Veronica, includes a clock circuit and a CPU. The breadboard is shown.



the one: *Getting Started in Electronics*.

That book was my bible for many years. I remember hacking a remote-control truck to have headlights and speed control. I remember building a working guillotine for a school project about the French Revolution. It was 4' tall and genuinely dangerous. I carried it on the bus and demonstrated it on bourgeoisie bananas in class. I don't imagine kids would get away with that today.

More recently, my interest in hacking was probably rekindled with the simple act of replacing the "non-user-serviceable" battery in a very expensive toothbrush (see "Toothbrush Repair," <http://quinndunki.com/blondihacks/?p=200>). That was five years ago, and I'm still using that toothbrush today. To me, that's the purest essence of hacking right there—fixing the one weakness in a product that would have otherwise halved its useful life.

NAN: Are you currently working on or planning any projects?

QUINN: My biggest hobby project recently has been my homebrew computer, Veronica

(see "Veronica," <http://quinndunki.com/blondihacks/?p=680>). The Apple II I mentioned was very formative for me, and [Apple Computer founder] Steve Wozniak was a bit of a hero figure. My whole life, I wanted to know how one person could just sit down and create something like that.


A couple of years ago, with no formal training in electrical engineering, I decided to see if I could do it. Veronica is the result, and it was the fulfillment of a lifelong goal to build a functioning, usable, 8-bit computer from scratch, complete with video graphics array (VGA) bitmapped video, a keyboard, game controllers, and built-in Pong.

Today, my most active project is repairing, restoring, and modifying an early 1990s Bally/Williams pinball machine called Johnny Mnemonic (see "Johnny," <http://quinndunki.com/blondihacks/?p=1559>). Anyone who likes hacking should be into pinball machines. They are wonderlands of mechanical systems, electronics, software, and game theory all rolled into one. They are also bottomless pits of hacking and tinkering potential (not to mention money pits and time sinks).

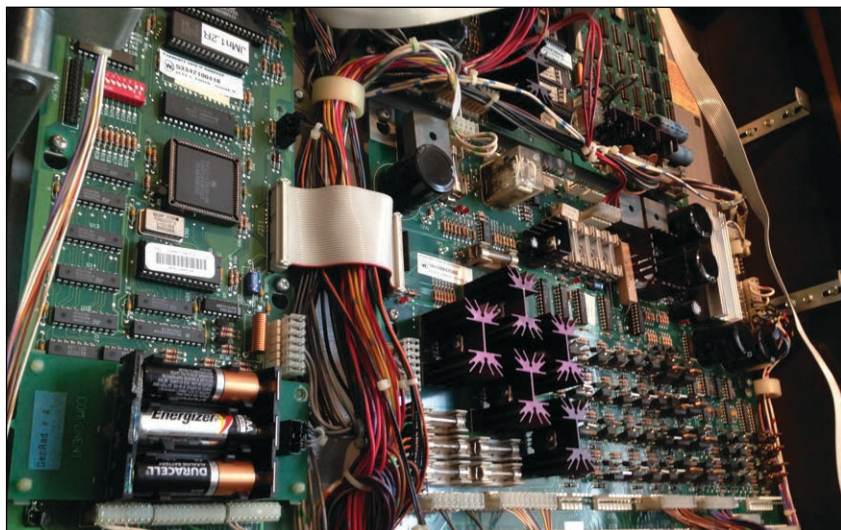
Another big ongoing project is our 24 Hours of LeMons (www.24hoursoflemons.com) race team. The short version is that it's a (very) low-budget form of endurance auto racing that involves a whole lot of hacking of all kinds. That's probably an entire interview unto itself, but I feel I should at least mention it, since it's such a big part of my hacking time.

NAN: What do you consider to be the "next big thing" in the industry?

QUINN: One "big idea" that has been put forward that I'm excited about is the notion of hackerspaces replacing public libraries. The Internet is gradually replacing the role of pure information access that libraries have served. It has been suggested that access to high-end technology creation tools is the next such area where playing field leveling is required. Anyone wanting to improve their station in life by executing their ideas in a high-tech world will need access to CNC machines, 3-D printers, machine tools, high-end computer-aided design (CAD), video production software, and so forth.

Libraries are generally well located and already equipped with things such as fire exits, sprinkler systems, and commercial-grade electrical. Converting some libraries into hackerspaces sounds to me like a terrific use of public funds. It's a bit "pie in the sky," but places like Edmonton in Alberta, Canada, and North Logan, UT, are already experimenting with the idea. This is like hacking democracy itself, and I love it. 

Quinn is in the process of hacking a Bally/Williams Johnny Mnemonic pinball machine. This photo shows the machine's circuitry.



PRODUCT NEWS

REAL-TIME PROCESSING FOR PCIe DIGITIZERS

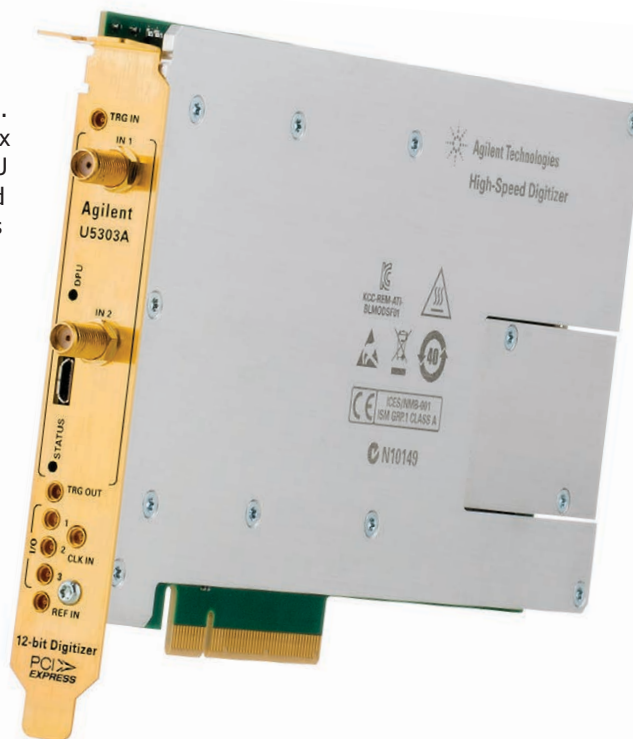
The **U5303A** digitizer and the **U5340A FPGA** development kit are recent enhancements to Agilent Technologies's PCI Express (PCIe) high-speed digitizers. The products add next-generation real-time peak detection functionalities to the PCIe devices.

The U5303A 12-bit PCIe digitizer features programmable on-board processing. It offers high performance in a small footprint, making it an ideal platform for many commercial, industrial, and aerospace and defense embedded systems. A data processing unit (DPU) based on the Xilinx Virtex-6 FPGA is at the heart of the U5303A. The DPU controls the module functionality, data flow, and real-time signal processing. This feature enables data reduction and storage to be carried out at the digitizer level, minimizing transfer volumes and accelerating analysis.

The U5340A FPGA development kit is designed to help companies and researchers protect their IP signal-processing algorithms. The FPGA kit enables integration of an advanced real-time signal processing algorithm within Agilent Technologies's high-speed digitizers. The U5340A FPGA features high-speed medical imaging, analytical time-of-flight, lidar ranging, non-destructive testing, and a direct interface to digitizer hardware elements (e.g., the ADC, clock manager, and memory blocks). The FPGA kit includes a library of building blocks—from basic gates to dual-port RAM—a set of IP cores, and ready-to-use scripts that handle all aspects of the build flow.

Contact Agilent Technologies for pricing.

Agilent Technologies, Inc.
www.agilent.com



PROGRAMMABLE LOGIC CONTROLLER BOARD

The **SmartTILE** (Smart TRi Integrated Logic Engine) is a programmable logic controller CPU board that plugs onto a carrier I/O board. The board integrates a 32-bit CPU,

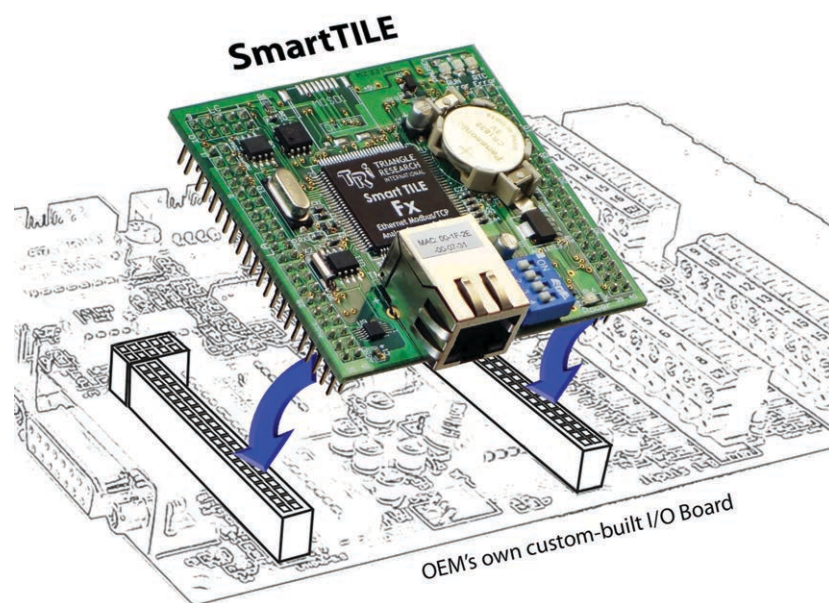
ferroelectric RAM (FRAM) and flash memory, a battery-backed real-time clock, and an Ethernet port on board. Its digital, analog, and serial I/O signals are brought to a user's carrier board via three sets of header pins.

All critical components are already built-in on board. You just design a simple carrier PCB that contains a D/A circuit that interfaces the SmartTILE's low-voltage signals to real-world voltages and currents (e.g., 24, 120, or 240 V).

The SmartTILE-Fx provides 16 digital inputs, 16 digital outputs (5-V CMOS logic level), eight analog inputs, and four analog outputs and can be expanded to 128 digital inputs and 128 digital outputs. The controller board -Fx provides three channels of serial ports that can interface to RS-232, RS-485, or even wireless radio. An I²C port is also available, enabling designers to interface to specialty ICs that support I²C bus.

Contact Triangle Research International for pricing.

Triangle Research International, Inc.
www.triplc.com



PRODUCT NEWS

“NO OPTO” SYNCHRONOUS FORWARD CONTROLLER

The **LT3752/LT3752-1** is a high-input voltage-capable synchronous forward controller with an active clamp transformer reset. A controlled V_{OUT} start-up and shut-down is maintained with an integrated housekeeping controller to bias the primary and secondary ICs. The internal bias generation also reduces the main power transformer's complexity and size by avoiding the need for extra windings to create bias supplies.

The LT3752 operates over a 6.5-to-100-V input voltage range. The LT3752-1 is well suited for hybrid vehicle (HV) and hybrid electric vehicle (HEV) applications. For up to 400-V inputs and greater, it enables RC start-up from the input voltage with the maximum voltage limited only by the choice of external components.

A $\pm 5\%$ output voltage regulation can be attained without using an optocoupler. An optocoupler can be used to obtain $\pm 1.5\%$ output voltage regulation. The LT3752/-1 uses a pulse transformer to send a control signal to a secondary-side MOSFET driver for the synchronous rectification timing. It can also be used in self-driven applications where the power transformer pulses control the secondary-side MOSFETs. With the LT3752/-1, secondary-side ICs no longer require start-up circuitry to operate when the output voltage is 0 V, which enables a controlled V_{OUT} start-up.

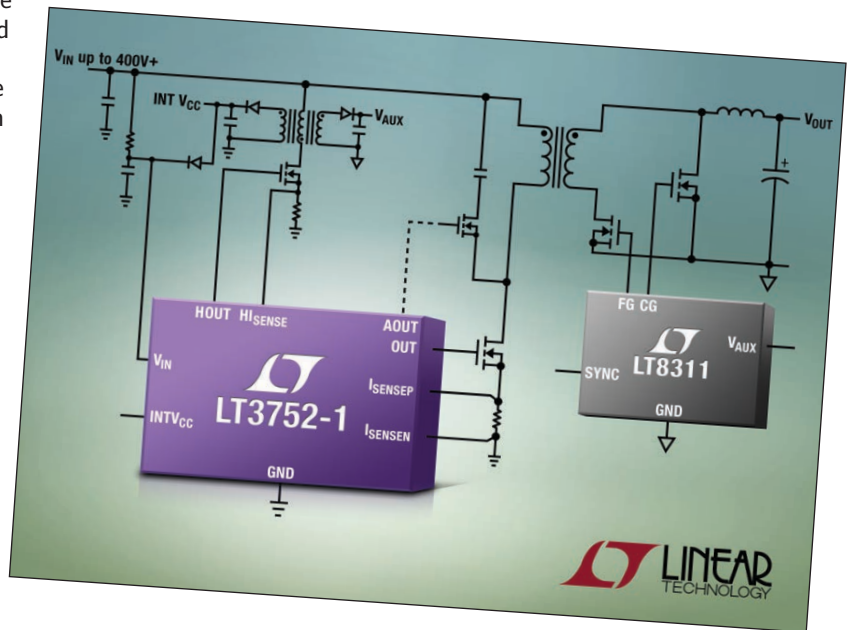
The LT3752/-1 has a programmable 100-to-500-kHz operating switching frequency. It can be synchronized to an external clock, enabling a range of output inductor values and transformer sizes to be used.

The LT3752/-1 is available in a TSSOP-38

package with several pins removed for high-voltage spacing. The LT3752/-1 E- and I-grade versions function from a -40°C -to- 125°C junction temperature. The LT3752/-1 H-grade version functions from a -40°C -to- 150°C operating junction temperature. The LT3752/-1 MP-grade version functions from -55°C -to- 150°C operating junction temperature.

The LT3752/LT3752-1 costs **\$3.39** in 1,000-piece quantities.

Linear Technology Corp.
www.linear.com



ULTRA-COMPACT ULTRASONIC SENSOR SERIES

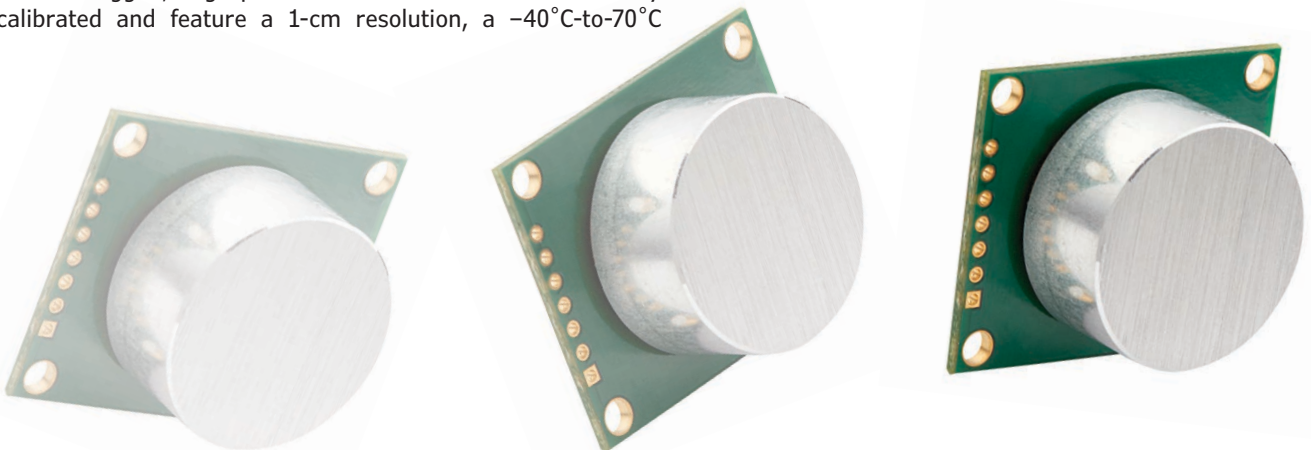
The **UCXL-MaxSonar-WR** series of sensors are flexible, OEM-customizable products that can be integrated into a system with MaxBotix's horns or flush-mounted into an existing housing. Mounting design recommendations are provided through MaxBotix's 3-D CAD models (available in multiple formats) to facilitate your design process. The sensor layout offers four conveniently placed mounting holes for design flexibility.

The rugged, high-performance sensors are individually calibrated and feature a 1-cm resolution, a -40°C -to- 70°C

operational temperature range, real-time automatic calibration (voltage, humidity, and ambient noise), 200,000-plus h mean time between failures (MTBF), and an operational 3-to-5.5-V voltage range with a low 3.4-mA average current requirement.

Contact MaxBotix for pricing.

MaxBotix, Inc.
www.maxbotix.com





Stronger than ever

IAR Embedded Workbench for ARM

Outstanding speed optimizations enable IAR Embedded Workbench to generate faster code than ever before. With the shortest possible execution times it is the ultimate choice for developing low-power applications. The world-leading C/C++ compiler and debugger tool suite, with the broadest MCU support, is now even more powerful.

www.iar.com



PRODUCT NEWS

LOW-POWER REMOTE-CONTROL TRANSCEIVERS

The **TT Series** remote-control transceiver is designed for bidirectional, long-range, remote-control applications. The module includes an optimized frequency-hopping spread spectrum (FHSS) RF transceiver and an integrated remote-control transcoder.

The FHSS is capable of reaching more than 2 miles in typical line of sight (LoS) environments with 0-dB gain antennas. An amplified version increases the output power from 12.5 to 23.5 dBm, boosting the range to more than 8 miles in LoS environments with 0-dB antennas.

The TT Series transceiver features best-in-class receive sensitivity (up to -111 dBm) and low power consumption (only 19.2 mA in Receive mode and 36 mA in Transmit mode at 12.5 dBm). The initial version operates in the 902-to-928-Hz frequency band for North America and South America.

The transceiver is housed in a compact reflow-compatible surface-mount technology (SMT) package. It doesn't require any external RF components except an antenna, which simplifies integration and reduces assembly costs.

Programming is not required for basic operation. The transceiver's primary settings are hardware-selectable, which eliminates the need for an external microcontroller or other digital interface. Eight status lines can be set up in any combination of inputs and outputs to transfer button or contact states. A selectable acknowledgment indicates that the transmission was successfully received. For advanced features, a UART interface provides optional software configuration.

A simple pairing operation configures two modules to operate together. A single button press on each side causes the modules to automatically swap their 32-bit addresses and store them in nonvolatile memory. It can be configured to automatically send

an acknowledgment to the transmitting unit either after receiving a command or with external circuitry when an action has taken place. An optional external processor can send two data bytes with the acknowledgement.

The TT Series transceiver module is available as part of Linx Technologies's master development system, which comes with two development boards for benchmarking and prototyping. Each board is populated with a transceiver, two remote-control development boards, and programming boards. The system also includes antennas, a daughterboard with a USB interface, demonstration software, extra modules, and connectors.

Contact Linx Technologies for pricing.

Linx Technologies
www.linxtechnologies.com



COM EXPRESS MODULE WITH 2ND-GENERATION INTEL CORE

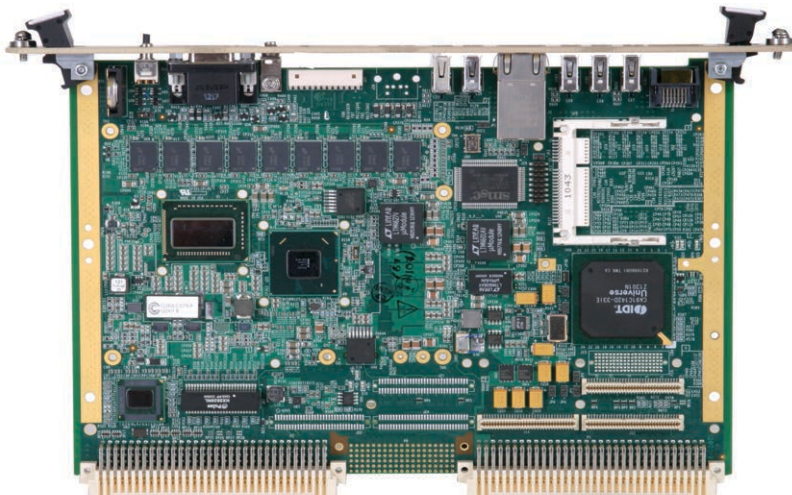
The **CPU-162-14** is a high-performance COM Express module based on the Intel Core i7 Processor. The module is

designed to work in harsh environments. It includes features to provide extra resilience to vibrations, making it well suited for transportation applications.

The CPU-162-14 module includes: extended temperature versions for -40° -to- 85° operation; direct-mounted RAM and a CPU to withstand stress and vibration; up to 8 GB double data rate type three synchronous dynamic RAM (DDR3 SDRAM); and three video ports including video graphics array (VGA), Intel's Serial Digital Video Out (SDVO), and low-voltage differential signaling (LVDS).

Contact Dynatem for pricing.

Dynatem, Inc.
www.dynatem.com



PRODUCT NEWS

GIGABIT ETHERNET DESIGNS

Würth Electronics Midcom and Lantiq recently announced **The Evaluation Kit**, a jointly developed demonstration kit. The kit enables users to easily add Ethernet hardware to an application or device and provides information to help users understand the demands of an Ethernet hardware design.

The Evaluation Kit includes an easy-to-use, 1-Gbps demonstration board. The 54-mm x 92-mm credit card-sized demonstration board is powered by USB. The board plugs into PCs and provides up to 1-Gbps bidirectional data rates.

The Evaluation Kit costs approximately **\$175**.

Würth Electronics Midcom, Inc.
www.we-online.com

Lantiq
www.lantiq.com



STAND-ALONE DATA LOGGER

The **DI-160** is a stand-alone event, state, and count data logger that features four programmable measurement modes. The data logger enables you to determine when events occur, the total number of events, and the period of time in between events. It can count parts by monitoring a proximity sensor's pulse output, or determine a machine's downtime by monitoring AC power.

The DI-160 includes eight channels. Four ± 300 -VDC/peak AC isolated channels can accommodate high-level DC voltage signals, pulse inputs up to 2 kHz, or AC line voltage. Four ± 30 -VDC/peak AC non-isolated channels (pulled high) enable you to monitor lower-level DC voltages, TTL-level, signals or switch closures.

You can use DATAQ's Event Recorder, which is included with the data logger, to set-up software, enable/disable channels, select measurement modes on a channel-by-channel basis, and choose one of 21 sample intervals, ranging from 1 s to 24 h. Data is stored to a removable SD memory card in CSV format, enabling up to 500 days of continuous recording and easy viewing in Microsoft Excel.

The DI-160's AC channels provide channel-to-channel and input-to-output isolation up to 500 VDC (± 250 -V peak AC) and have a 4-V trigger threshold. The low-voltage channels are protected up to ± 30 VDC/peak AC and trigger at 2.5 V.

A built-in rechargeable battery acts as a "bridge" when disconnecting the data logger from a PC and connecting it to the USB power supply. Three LEDs indicate when the DI-160 is actively acquiring data, when the unit is connected via USB to a PC (or the included AC power supply), and the battery's charge state.

A push button enables you to start and stop recording to the SD memory card.

The DI-160 has four selectable measurement modes. State mode determines an event's duration. Event mode detects a single change of state (within a sample interval). High-Speed (HS) Counter mode yields the total number of state changes within a sample interval. AC Counter mode counts the number of times AC power turns on/off within a sample interval.

The DI-160 costs **\$299** and includes a mini screwdriver, a 2-GB SD memory card, an AC power supply, and a mini-USB cable. The DATAQ Event Recorder software is available for free download.

DATAQ Instruments, Inc.
www.dataq.com



PRODUCT NEWS

ARBITRARY WAVEFORM GENERATOR

The **AWG18** is an arbitrary waveform generator designed for use with Applicos's ATX-series of test systems. The waveform generator features an 18-bit resolution at a 300-megasamples-per-second (MSPS) data rate and oversampling at a 600-MSPS or 1.2-gigasamples-per-second (GSPS) rate.

Testing analog systems with high-speed 14- and 16-bit data converters requires extremely clean signals. The traditional approach of filtering away the harmonics is insufficient when dealing with a high signal-to-noise ratio (SNR), maintaining a high spurious-free dynamic range (SFDR), or when a low "close-in carrier noise" is preferred. In these cases, the extra precision from an 18-bit signal value can create a reliable test-stand operation and reduce random failures.

Many applications need to use signals other than sine waves to make additional time domain measurements. The AWG18 has two signal paths to accommodate this. One path starts at DC for time domain and general-purpose measurements that require high-level accuracy. The other path runs from 10 to 100 MHz and is optimized for dynamic signal generation in this frequency range.

The typical total harmonic distortion (THD) for frequencies up to 50 MHz is above 100 dBc.

Contact Applicos for pricing.

Applicos BV
www.applicos.com



16-BIT DIGITIZER

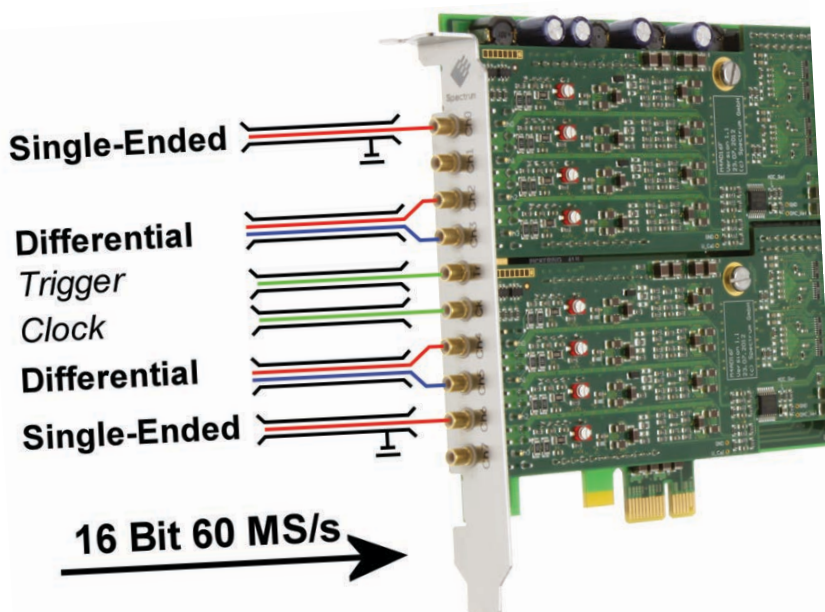
The **M2i.4960** and the **M2i.4961** mid-speed 16-bit digitizers are available for PCI/PCI-X and PCI Express (PCIe). The devices offer two or four synchronous channels with a 60 megasamples-per-second (MSPS) speed and a 30-MHz bandwidth.

The channels can be individually switched between single-ended and true differential input mode; therefore, single-ended and differential signals can be simultaneously acquired with one digitizer. Each input channel includes on-board calibration. The channels can be software programmed for proper termination, user offset, and input range.

The devices' acquisition modes include segmented acquisition, gated acquisition, or streaming mode. The digitizers also feature a versatile clock and trigger section, making them suitable for a variety of different applications. Multiple cards can be internally synchronized to obtain more synchronous channels or to directly synchronize to arbitrary and digital waveform generators or digital waveform capture cards.

A digital input option enables up to 32 synchronous digital input channels to be acquired by multiplexing them into the analog data in different ways. Each of the 16 digital inputs can completely replace one analog channel or each of the 2/4 digital inputs can be stored together with the A/D sample by reducing its resolution.

Contact Spectrum for pricing.



Spectrum GmbH
www.spectrum-instrumentation.com

CLIENT PROFILE

Measurement Computing Corp.

www.mccdaq.com

10 Commerce Way, Norton, MA 02766

CONTACT: Dan Mandill, dan.mandill@mccdaq.com

EMBEDDED PRODUCTS: Measurement Computing Corp. (MCC) sells measurement-grade data acquisition (DAQ) devices for Windows, Linux, and Android applications. MCC provides USB, Ethernet, wireless, and PCI/PCI Express solutions to embedded developers and engineers seeking measurement-grade inputs, full-featured software drivers, and free technical support.

MCC's latest software driver, Universal Library (UL) for Android, supports several DAQ devices on the Android platform. The UL for Android driver delivers timed analog input along with single-point analog input, analog output, 5-V digital I/O, and counters. Used with MCC DAQ boards, UL for Android provides application developers with high-quality I/O on systems including tablets, smartphones, and SBCs.

FEATURED PRODUCT: The BTH-1208LS is a wireless multifunction DAQ board that is supported by Windows and Android. With easy-to-set-up Bluetooth or USB connectivity, the board offers eight analog inputs that can be continuously

acquired by the host at up to 1.2 kilosamples per second (kSps) or up to 50 kSps to local memory.

The BTH-1208LS board also provides two analog outputs, eight digital I/Os, and a counter, and runs on rechargeable batteries. An OEM version is available without batteries and with standard headers.

EXCLUSIVE OFFER: For a limited time, *Circuit Cellar* readers will receive a 20% discount off the BTH-1208LS. Visit <http://bit.ly/1I2TzQ> and enter the promo code CCBTH at checkout.

This space is where Circuit Cellar enables clients to present readers useful information, special deals, and more.



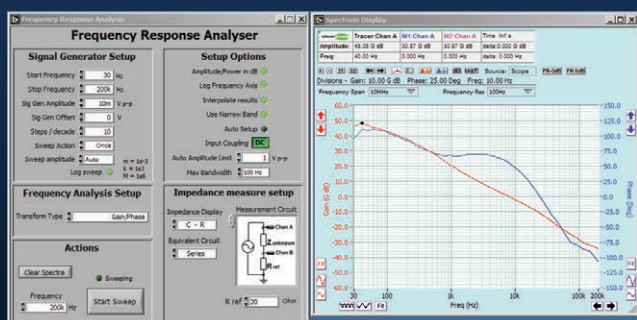
INDUSTRY & ENTERPRISE

Frequency Response Analysis

Use the Cleverscope FRA panel to easily auto plot Gain/Phase, Impedance, Capacitance or Inductance vs Frequency. Display the Gain and Phase Margin.

Easy As, with Cleverscope.

See our FRA tutorial video to show you how to verify your operating power supply or amplifier design. Check the impedance of your DC buses. Verify magnetics you have wound. 80 dB dynamic range! 0 - 65 MHz isolated Sig Gen.



Streaming 100 G samples to disk • **Frequency Response Analysis** • Protocol Analysis • Symbolic Math • Matlab Interface • 80 dB dynamic range • 100 MHz Bandwidth • Tracking Zoom • 0-65MHz isolated sig gen • Video Tutorials



CS328A-FRA
14 Bit MSO



www.cleverscope.com

Industrial Temperature SBC

iPac-9x25

- Atmel 400 Mhz Processor
- 128MB DDR2 RAM, 4GB eMMC
- 16MB Serial Data Flash, Micro SD
- 36 General Purpose Digital I/O lines
- 8 High Drive Digital Outputs
- 1x USB 2.0 (High-Speed) Host
- 1x USB (Full-Speed) Host
- 1x USB 2.0 (High-Speed) Device Port
- 3x RS232, 1x RS232/422/485
- 2x 10/100 Ethernet, 1x CAN Bus
- 1x I2C Port & 1x SPI Port
- Up to 7 channels of 10 bit A/D
- Up to 4 16-bit PWMs
- Industrial Temp. -40C to +85C



3.10 KERNEL

Designed and manufactured in the USA, the iPac-9X25 is a web enabled microcontroller with the ability to run an embedded server and to display the current monitored or logged data. The web connection is available via two 10/100 Base T Ethernet ports or 802.11 wireless wifi networking when using the proper Linux modules and adapters. This Microcontroller has all connectors brought out as headers on a board and has the same footprint of a standard PC/104 module at 3.77" x 3.54". The iPac-9X25 is perfectly suited for Industrial Temperature Embedded Data Acquisition and Control applications. Pricing for Qty 1 is \$198

www.emacinc.com/sales/ipac14

Since 1985
OVER
29
YEARS OF
SINGLE BOARD
SOLUTIONS

EMAC, inc.
EQUIPMENT MONITOR AND CONTROL

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

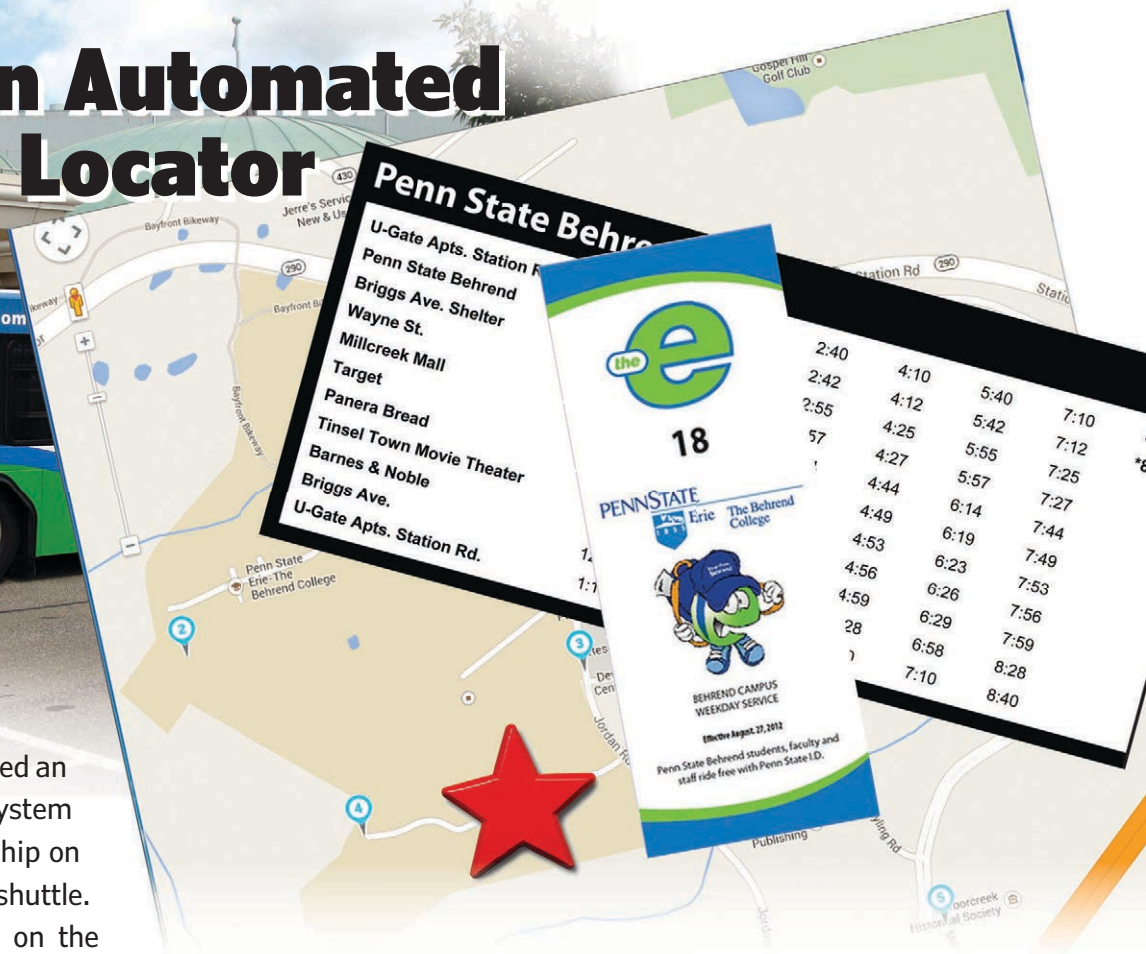
Build an Automated Vehicle Locator

FEATURES



This design team created an online bus-tracking system to help increase ridership on a university campus shuttle. A mobile GPS tracker on the bus communicates over a radio link to a base station. The system reliably and accurately predicts the bus location even when the vehicle is out of radio contact range.

By Chris Coulston, Daniel Hankewycz, and Austin Kelleher (US)



The continued expansion of the Penn State Erie campus in Erie, PA, prompted a shuttle bus service provided by the Erie Metro Transit Authority (EMTA). The shuttle services eight stops around the campus on a loop that covers approximately 3 miles in 20 min. Unfortunately, ridership is low, partly because of the bus' unpredictable schedule. Riders want to know where the bus is and when it will arrive at a stop.

In addition to the environmental benefits, increased bus ridership would reduce the growing traffic problems encountered between class periods. To address low ridership, we constructed an automated vehicle locator to track our campus shuttle and provide accurate estimates of when the bus will arrive at each stop.

THE BIG PICTURE

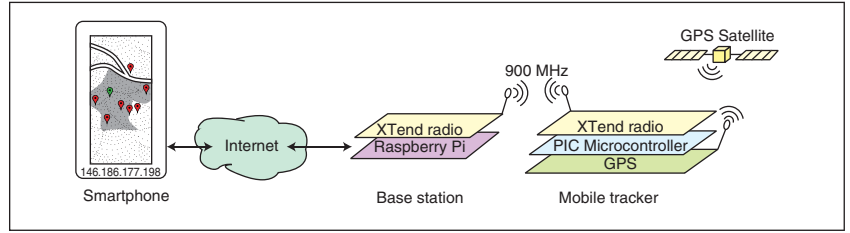
Before we dive into the bus tracker's details, we'll provide a broad overview of the hardware and software used in this project. If you don't entirely understand how the system works after reading this section, don't worry, that's what the rest of the article is for! **Figure 1** shows the bus tracker's hardware, which consists of three components: the user's smartphone, the base station placed at a fixed location on campus, and the mobile tracker that rides around on the bus.

Early on, we decided against a cellular-based solution (think cell phone) as the mobile tracker. While this concept would have benefited from wide-ranging cellular coverage, it would have incurred monthly

cellar network access fees. **Figure 1** shows the final concept, which utilizes a 900-MHz radio link between the mobile tracker and the base station.

Figure 2 shows the software architecture running on the hardware from **Figure 1**. When the user's smartphone loads the bus tracker webpage, the JavaScript on the page instructs the user's web browser to use the Google Maps JavaScript API to load the campus map. The smartphone also makes an XMLHttpRequests request for a file on the server (stamp.txt) containing the bus' current location and breadcrumb index (more on this later).

This information along with data about the bus stops is used to position the bus icon on the map, determine the bus' next stop, and predict the bus' arrival time at each of the seven bus stops. The bus' location contained in stamp.txt is generated by a GPS receiver (EM-408) in the form of an NMEA string. This



string is sent to a microcontroller and then parsed. When the microcontroller receives a request for the bus' location, it formats a message and sends it over the 900-MHz radio link. The base station compares the bus position against a canonical tour of campus (breadcrumb) and writes the best match to stamp.txt.

FIGURE 1

The bus tracking system includes a Digi International XTend radio, a Microchip Technology PIC18F26K22 microcontroller, and a Raspberry Pi single-board computer.

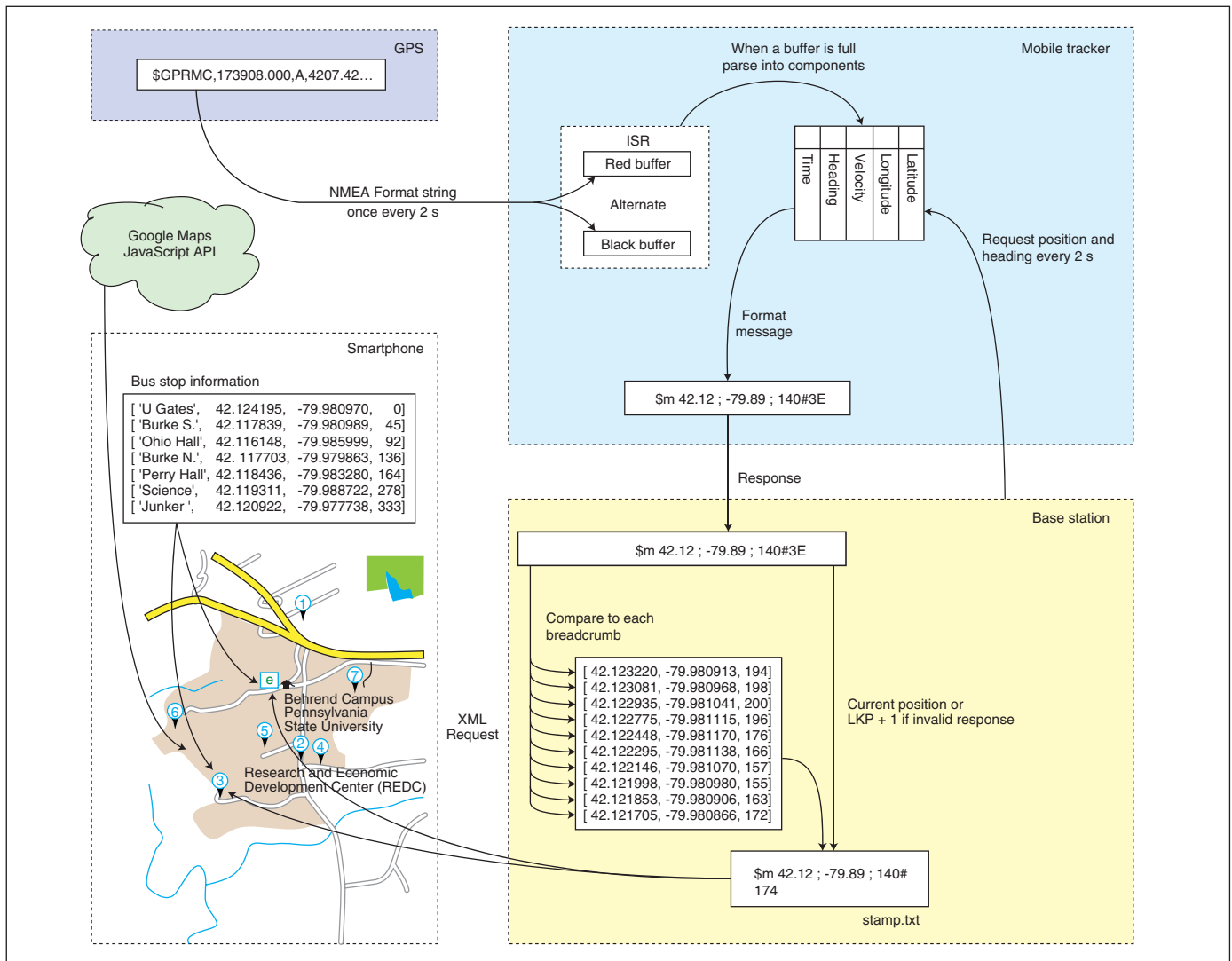


FIGURE 2 The bus tracker's software architecture includes a GPS, the mobile tracker, a smartphone, and the base station.

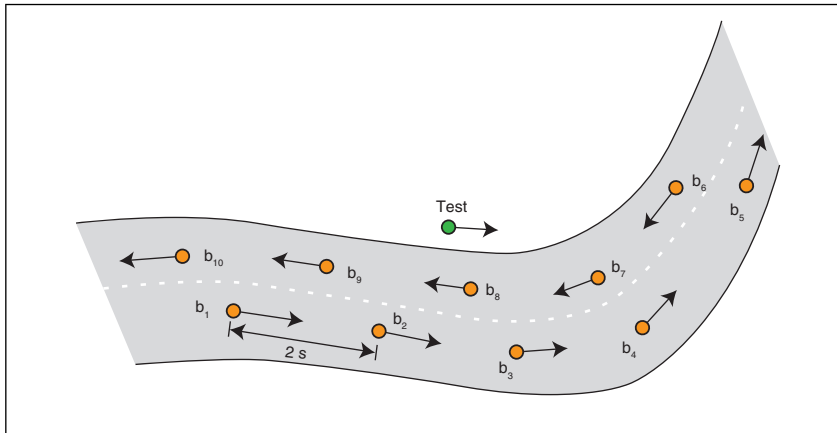


FIGURE 3 Breadcrumbs (b_1 through b_{10}) containing the bus' position and orientation information were taken every 2 s during a test-run campus tour.

FIGURE 4 The mobile tracker includes a Microchip Technology PIC18F26K22 microcontroller, a Micrel MIC5205 regulator, a Digi International XTend RF module, and a Texas Instruments TXS0102 bidirectional translator.

is called "breadcrumbs" because, like the breadcrumbs dropped by Hansel and Gretel in the eponymously named story, we hope they will help us find our way around campus. **Figure 3** shows a set of breadcrumbs (b_1 through b_{10}), which were collected as the bus traveled out and back along the same road.

The decision to collect breadcrumbs proved fortuitous as they serve an important role in each of the three hardware components shown in **Figure 1**. Now that you have the big picture, we'll dive into the design decisions that converted this concept into a robust bus-tracking system.

MOBILE TRACKER

The bus houses the mobile tracker. **Figure 4** shows the schematic, which is deceptively simple. What you see is the third iteration of the mobile tracker hardware.

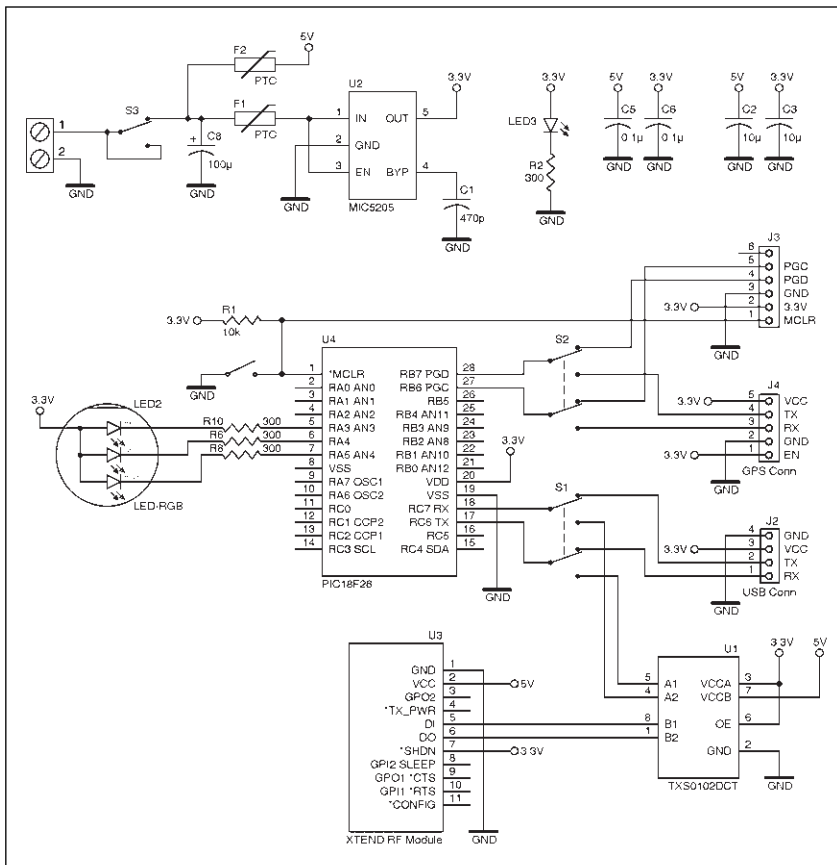
An important starting point in the design was how to step down the bus' 12-V supply to the 5-V required by our circuit. In terms of hardware, the best decision we made was to abandon the idea of trying to integrate a 12-to-5-V converter onto the mobile tracker PCB. Instead we purchased a \$40 CUI VYB15W-T DC-DC converter and fed the mobile tracker 5-V inputs.

Using an off-the-shelf isolated power supply reassured the EMTA that our circuit would not damage their bus and helped convince us that power spikes on the bus' mains power would not damage our circuit. A pair of PTC thermal fuses between the power supply and circuitry provided additional safeguarding.

We used Micrel's MIC5205 regulator to step down the 5 V for the 3.3-V GPS receiver, which easily supplied its peak 80 mA. Since we ran a Digi International XTend radio at 5 V for the best range, we ended up with mixed voltage signals. We used a Texas Instruments TXS0102 bidirectional voltage-level translator, which handles voltage-interfacing duties between the 5-V radio and the 3.3-V microcontroller.

We selected Microchip Technology's PIC18F26K22 because it has two hardware serial ports, enabling it to simultaneously communicate with the GPS module and the radio modem when the bus is traveling around campus. We placed two switches in front of the serial ports. One switch toggles between the GPS module and the Microchip Technology PICKit 3 programming pins, which are necessary to program the microcontroller. The second switch toggles between the radio and a header connected to a PC serial port (via a Future Technology Devices FT232 USB-to-serial bridge). This is useful when debugging at your desk. An RGB LED in a compact PLCC4 package provides state information about the mobile tracker.

The XTend RF modules are the big brothers to Digi International's popular XBee series. These radios come with an impressive 1 W of transmitting power over a 900-MHz frequency, enabling ranges up to a mile in our heavily wooded campus environment. The radios use a standard serial interface requiring three connections: TX, RX, and ground. They are simple to set up. You just drop them into the Command mode, set the module's source and destination addresses, store this configuration in flash memory, and exit. You never have to deal with them again.



CadSoft
presents

EAGLE

V6

Felicitas PCBsIm

the new simulation tool for signal integrity tests*:

- Signal integrity testing before the layout to avoid redesign
- Integration in EAGLE and intuitive operation
 - Accurate simulation results
 - Support directly from manufacturer

*developed by Felicitas Customized Engineering

www.cadsoftusa.com

25 years CadSoft



1988 - 2013

LISTING 1

The code shows the mobile tracker's executive function menu.

```

----- System Status -----
Tour 0
GPS src  USART2
Sim type clean
Output USART1
----- WDT configuration -----
WDT: Enabled
WDT pre: 1:4096
----- GPS Status -----
Time 133020.000
Date 101013
Lat:Lon 42.118458:-079.982588
Velocity 13
Heading: 62
GPS signal:Locked
----- Query GPS -----
?: help  s: gps String t: Time
p: Position l: Latitudeg: lonGitude
v: Velocity m: map h: Heading
i: GPS signal statusd: Date
----- Change system mode -----
o: tOggle source of NMEA strings between simulation and USART2
c: Change USART used by printf
k: Kick simuation ahead 30 seconds
f: conFigure GPS to 9600 baud RMC
F: configure GPS to Factory defaults
R: Reboot the system

```

```

xbee = serial.Serial( port=/dev/ttyAMA0,      baudrate=9600, xonxoff=False,
                    dsrdrtr=False,          writeTimeout=0,
                    timeout=0.5,          stopbits=serial.STOPBITS_ONE,
                    rtscts=False,         parity=serial.PARITY_NONE)

while True:
    xbee.open()                                # open the serial port
    fcntl.lockf(xbee,fcntl.LOCK_EX)           # blocking lock for exclusive access
    xbee.flush()                               # remove any remnant junk
    xbee.write('m')                           # query mobile position
    inLine = xbee.readline()                  # read response and then
    fcntl.lockf(xbee,fcntl.LOCK_UN)          # unlock the port for another user
    xbee.close()                              # and then close the port

    minIndex = closestBreadCrumb(inLine)      # index of closest breadcrumb
    f = open("/var/www/remote/stamp.txt", "w")
    if (minIndex != -1):                      # error code = -1
        f.write(inLine[0:string.find(inLine,'#')+1]+'\\n' )
        f.write(str(minIndex) + '\\n')
        lkpIndex = minIndex                  # refresh Last Know Position
    else:
        lkpIndex += 1;
        f.write("$m " + str(breadCrumb[lkpIndex][1]))
        f.write(" ; " + str(breadCrumb[lkpIndex][2]))
        f.write(" ; " + str(breadCrumb[lkpIndex][3]) + "#\\n")
        f.write(str(lkpIndex)+'\\n')
    time.sleep(2.0)

```

LISTING 2

Python code runs on the base station to query the mobile tracker.

sensors

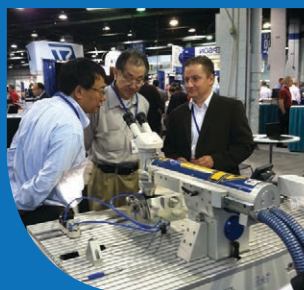
expo & conference

www.sensorsexpo.com

June 24-26, 2014

Donald E. Stephens Convention Center • Rosemont, IL

Sensing Technologies Driving Tomorrow's Solutions

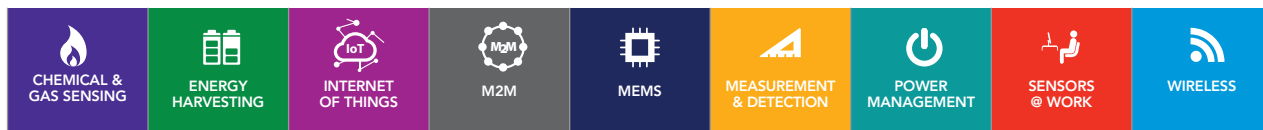


**SPECIAL
Subscriber
Discount!**

Register with code **A303C**
for \$50 off Gold and Main
Conference Passes.*

What's Happening in 2014:

Tracks



Plus+

- Full-day Pre-Conference Symposia
- Technology Pavilions on the Expo Floor
- **Internet of Things**
- **Energy Harvesting**
- **MEMS**
- **Wireless**
- **High Performance Computing**

New Co-location with High Performance Computing Conference

- Best of Sensors Expo 2014 Awards Ceremony
- Networking Breakfasts
- Welcome Reception
- Sensors Magazine Live Theater
- And More!



Featuring Visionary Keynotes:



Reimagining Building Sensing and Control

Luigi Gentile Polese
Senior Engineer
National Renewable Energy Lab



Sensors, The Heart of Informatics

Henry M. Bzeih
Head of Infotainment & Telematics
Kia Motors America



Innovative Applications. Expert Instructors. Authoritative Content. Tomorrow's Solutions.

Register today to attend one of the world's largest and most important gatherings of engineers and scientists involved in the development and deployment of sensor systems.

Registration is open for Sensors 2014!

Sign up today for the best rates at www.sensorsexpo.com or call 800-496-9877.



#sensors14

OFFICIAL PUBLICATION:

sensors

INDUSTRY SPONSOR



CO-LOCATED WITH:



*Discount is off currently published rates. Cannot be combined with other offers or applied to previous registrations.

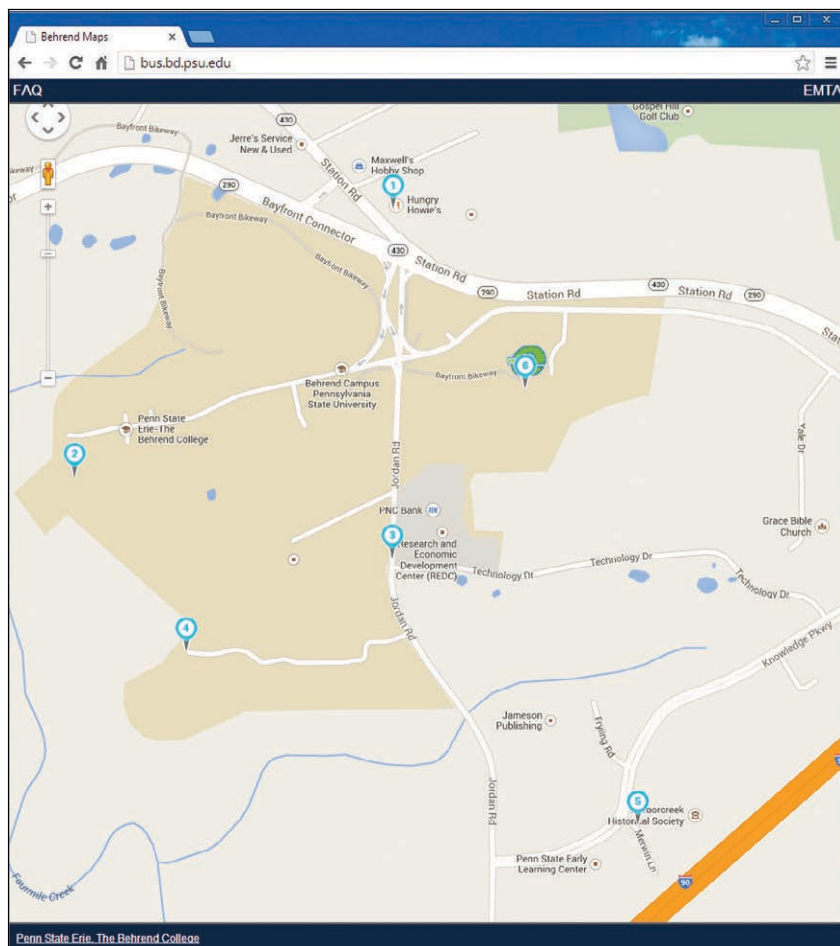


PHOTO 1

This is the bus tracker web application. Numbers are bus stops and the “e” (which is covered by the “6”) is the bus’ location.

Any character sent to the radio appears on the destination modem’s RX line.

The GPS receiver utilizes the CSR SiRFstarIII chipset, which is configured to output a recommended minimum specific (RMC) string every 2 s. An RMC string consists of several comma-delimited fields:

```
$GPRMC,173908.000,A,4207.4253,N,07958.8668,W,5.23,140.26,290513,,,*73
```

From left to right, they are the header identifying the type of string, the time (24-h format), the lock status (A = valid), the latitude, the north/south indicator, the longitude, the east/west indicator, the speed (knots), the heading, the date, and an XOR-based checksum.

Take care when interpreting the latitude and longitude as they are in a DDMM.MMMM format where D digits are degrees and the M digits are minutes. The Google Maps API expected latitude and longitude in DD.DDDDDD (decimal degree) format. The solution is to multiply the minute’s value by 100/60. Dividing it by 60 scales it to [1-0] and then multiplying it by 100 puts it in base 10 notation. The checksum is just the bitwise XOR of the characters following the \$ and preceding the *.

The mobile tracker’s firmware listens for commands over the serial port and generates appropriate replies. Commands are issued by the developer (e.g., ?, which generates the help menu shown in **Listing 1**) or by the base station. The most common command issued by the base station is m, which generates a semicolon-delimited string consisting of the bus’ latitude, longitude, and heading, followed by a checksum (e.g., \$m 42.121161 ; -079.982918 ; 256#4e).

Burning breadcrumbs into the mobile tracker’s flash memory proved to be a good design decision. With this capability, the mobile tracker can generate a simulated tour of campus while sitting on the lab bench.

BASE STATION

The base station consists of an XTend RF module connected to a Raspberry Pi’s serial port. The software running on the Raspberry Pi does everything from running a Nginx open-source web server to making requests for data from the mobile tracker.

From **Figure 1**, the only additional hardware associated with the base station is the 900-MHz XTend radio connected to the Raspberry Pi over a dedicated serial port on pins 8 (TX) and 10 (RX) of the Raspberry Pi’s GPIO header.

The only code that runs on the base station is the Python program, which periodically queries the mobile tracker to get the bus’ position and heading (see **Listing 2**). The program starts by configuring the serial port in the common 9600,8,N,1 mode. Next, the program is put into an infinite loop to query the mobile tracker’s position every 2 s.

The serial port is opened with exclusive access and closed each time it is used. This enabled different processes to issue commands to the mobile tracker (through the serial port) without the responses becoming garbled with one another. This situation occurs frequently during development any time diagnostics need to be performed on the mobile tracker while the code in **Listing 2** is executing. The `closestBreadcrumb(inLine)` function in **Listing 2** determines how far along the bus is in its route using the breadcrumb array from **Figure 3**. The input to this function is the current latitude, the longitude, and the bus’ heading (“test” in **Figure 3**). It returns the breadcrumb’s index with the minimum distance to the bus using:

$$\text{dist} = |\Delta\text{head}| + \sqrt{(111,320 \times \Delta\text{lat})^2 + (74,630 \times \Delta\text{lng})^2}$$

Some explanation about this equation is needed. Regardless of where you are on

```

/**
 * This function reads the contents of the stamp.txt file
 * @return a string containing latitude, longitude, heading
 */
function readGPS() {

    var xmlhttp;
    if (window.XMLHttpRequest) {
        xmlhttp=new XMLHttpRequest();
    } else {
        xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
    }

    xmlhttp.onreadystatechange=function() {
        if (xmlhttp.readyState==4 && xmlhttp.status==200)
            gpsString=xmlhttp.responseText;
    }

    xmlhttp.open("GET","./remote/stamp.txt",true);
    xmlhttp.send();
}

/**
 * This function updates the position of the bus and information available to users
 * @param refreshBusStops Boolean directing function to update arrival times at bus stops
 */
function updateMapMarkers(refreshBusStops){

    var cords = new Array;

    readGPS();
    setTimeout(function(){ // "returns" global gpsString
        // Since it takes a while to get a response
        cords = parsePosition(gpsString);
        bus.currLong = parseFloat(cords.pop());
        bus.currLat = parseFloat(cords.pop());
        bus.currHead = parseInt(cords.pop());
        bus.breadIndex = parseInt(cords.pop());
        if(refreshBusStops == "TRUE") refreshBusStopMarkers();
        refreshBusPosition();
    }, 300);
} // end

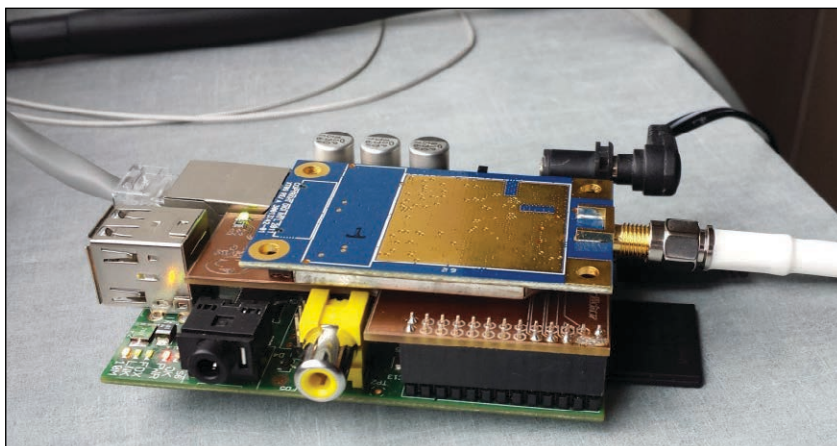
```

LISTING 3

This Java code reads the GPS and sets the bus stop markers' positions.

earth, a decimal degree of latitude (north/south) is always 111,320 m. This is not so with longitude. At our campus' longitude, which is 42.12° north, a decimal degree is $\sin(42.12) \times 111,320 = 74,630$ m. The square root term in the equation converts the difference between a breadcrumb's position and bus' position into a distance in meters. The heading is included in this calculation because without it, we would not be able to tell if the bus was traveling up the road toward the residence hall, or down the same road toward the engineering building.

For example, in **Figure 3** the point labeled

**PHOTO 2**

The base station includes an interface board, a Raspberry Pi, and a radio modem.

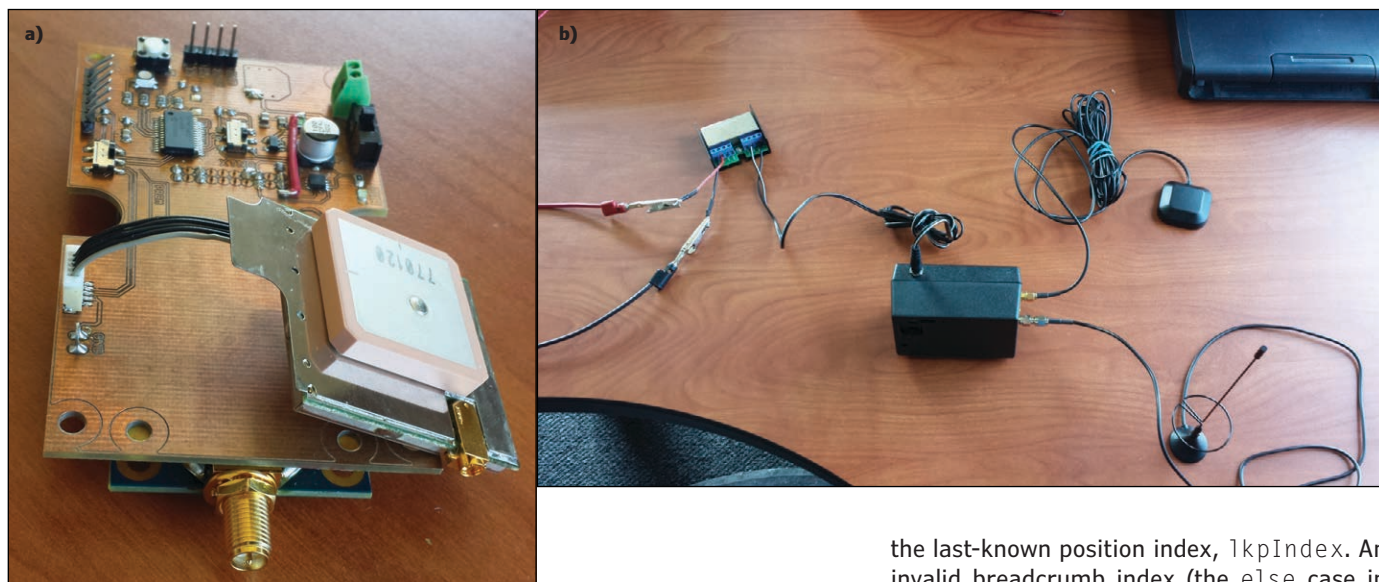


PHOTO 3

The mobile tracker unit (a) sits in the black case, ready for installation in the bus (b).

“test” is geographically closest to breadcrumb b8. However, including the heading, it shows that it really is closest to b2.

If the `closestBreadcrumb(inLine)` function returns a valid breadcrumb index (the “if” case in Listing 2), the bus’ location and heading along with the breadcrumb index are written to the file `stamp.txt`. In addition, this breadcrumb index is stored as

the last-known position index, `lkpIndex`. An invalid breadcrumb index (the `else` case in Listing 2) means the radio is out of the base station’s range.

When this happens, the last-known position index is incremented and the breadcrumb at that index is stored in `stamp.txt`, which creates an inferred position. This inferred bus position is updated every 2 s until the mobile tracker comes back in range of the base station, at which point the code in Listing 2 will resume storing the mobile tracker’s actual position in `stamp.txt`.

This technique of fusing an inferred position when radio contact is lost is beneficial to users. The user is assured that the system is properly functioning when he sees the bus moving normally on its route.

The drawback of this technique is that the bus position may “glitch” when transitioning from the inferred position back to its actual position. In practice, this technique works exceptionally well, surpassing our expectations. This is a good thing as the mobile tracker is out of radio contact with the base station during roughly 25% of its route.

SMARTPHONE

Photo 1 shows the user interface presented by the Bus Locator web page. The Raspberry Pi’s Nginx server delivers the user `index.php`, which directs the browser to load a campus map via the Google Maps JavaScript API and JavaScript code, which overlays the bus and bus stops.

Listing 3 shows two functions, `readGPS` and `updateMapMarkers`, which overlay the bus and bus stop icons on the campus map. The `readGPS` function uses Asynchronous JavaScript (AJAX) and XML to read the contents of the `stamp.txt` file generated by Listing 2. After creating a request object (`xmlhttp`) and sending a request to read `stamp.txt`, things get a little weird. The main

PROJECT FILES



circuitcellar.com/ccmaterials

RESOURCES

Nginx, <http://nginx.org>.

PySerial, <http://pyserial.sourceforge.net>.

Raspberry Pi, www.raspberrypi.org.

W3Schools, www.w3schools.com.

SOURCES

SiRFstarIII Chipset

CSR plc | www.csr.com

VYB15W-T DC-DC Converter

CUI, Inc. | www.cui.com

XTend and XBee RF modules

Digi International, Inc. | www.digi.com

LM7805 Linear regulator

Fairchild Semiconductor Corp. | www.fairchildsemi.com

FT232 USB-to-serial bridge

Future Technology Devices International, Ltd. | www.ftdichip.com

MIC5205 Regulator

Micrel, Inc. | www.micrel.com

PIC18F26K22 Microcontroller and PICkit 3 debugger

Microchip Technology, Inc. | www.microchip.com

TXS0102 Bidirectional voltage-level translator

Texas Instruments, Inc. | www.ti.com

processing thread breaks into two parts. One part runs `xmlhttp.onreadystatechange` and the other part continues executing the main thread. The `xmlhttp` object goes through five states on its way from being created, `readyState==0`, to being completed with a response ready, `readyState==4`. These state changes are monitored by the callback function associated with the `onreadystatechange` variable. When the response is ready, the `stamp.txt` file's entire contents are available through `xmlhttp.responseText`.

Since the thread that contains the XML request split from the main thread, the `updateMapMarkers()` function needs to reunite them to use `gpsString`. The `setTimeout` function does this by creating a 300-ms delay to read `stamp.txt` before executing its callback function. This function parses out the information contained in `stamp.txt`, occasionally refreshes the bus stop markers, and redraws the bus icon. These last two functions require information derived from the breadcrumb array.

The breadcrumb array was analyzed offline to determine the index in the array for each bus stop. This information is stored in the bus stop array. (Refer to the rightmost column in "Bus stop information" in **Figure 2**.)

The bus' estimated time of arrival (ETA) at a particular stop is two times the difference between the bus' current breadcrumb index, which is given by `bus.breadIndex` and the bus stop's breadcrumb index. The factor of 2 is introduced because there is 2 s between each breadcrumb. Each stop's ETA is provided whenever a user lingers over a particular bus stop. Breadcrumbs also help determine which stop the bus is headed toward. This is accomplished by taking the bus' breadcrumb index and finding the next highest bus stop breadcrumb index.

FINAL IMPLEMENTATION

Photo 2 and **Photo 3** show the final hardware implementation of the base station and mobile tracker. The base station radio is connected to an L-com 8-dBi flat-patch antenna (white cable) covering a roughly 120° arc. This is perfect for its location near Bus Stop 4 in **Photo 1**.

A Fairchild Semiconductor LM7805 linear regulator on the interface board provides auxiliary 5-V power for the radio. The mobile tracker is installed entirely inside the bus. Initial concerns about attenuation of the GPS and radio signals through the bus' fiberglass skin proved to be unmerited.


You can view the bus tracker's final implementation at <http://bus.bd.psu.edu> from 7:40 AM to 7:00 PM EST Monday through

ABOUT THE AUTHOR

Chris Coulston (coulston@psu.edu) holds a BA in Physics (Slippery Rock University), a BS and an MS in Computer Engineering (Penn State University), and a PhD in Computer Science (Penn State University). He has worked at Penn State Erie for 13 years as an Electrical and Computer Engineering professor and currently serves as the Department Chair of the Computer Science and Software Engineering department. His technical interests include embedded systems, computer graphics algorithms, and sensor design. Chris is also a *Circuit Cellar* project editor.

Daniel Hankewycz (djh5533@psu.edu) is a second-year Computer Engineering student at Penn State Erie. Daniel enjoys working with embedded systems, PCB design, and CNC technology.

Austin Kelleher (alk5492@psu.edu) is a second-year Computer Science student at Penn State Erie. His technical interests include cross-platform technologies, network security, and artificial intelligence.

Friday, from September to May. The system works remarkably well, providing reliable, accurate information about our campus bus. Best of all, it does this autonomously, with very little supervision on our part. It has worked so well we have received additional funding to add another base station to cover an extended campus route next year. 



Keil MDK-ARM

MDK-ARM™ is the complete development environment for ARM® Cortex™-M series microcontrollers.

- CAN Interface
- File System
- USB Host
- USB Device
- TCP/IP Networking
- GUI Library

ARM® www.keil.com/mdk
 +49 89 456040-20

© ARM Ltd. AD366 | 01.13

Bit Banging

UARTs are peripherals that move information from device to device. This article describes how to use the UART “protocol” to implement systems that transmit and receive data without a built-in peripheral. Implementing such a transmitter or receiver in software is called bit banging.

By Shlomo Engelberg (Israel)

There is no better way to understand a protocol than to implement it yourself from scratch. If you write code similar to what I describe in this article, you’ll have a good understanding of how signals are transmitted and received by a UART. Additionally, sometimes relatively powerful microprocessors do not have a built-in UART, and knowing how to implement one in software can save you from needing to add an external UART to your system. It can also reduce your parts count.

WHAT DOES “UART” MEAN?

UART stands for universal asynchronous receiver/transmitter. The last three words in the acronym are easy enough to understand. “Asynchronous” means that the transmitter and the receiver run on their own clocks. There is no need to run a wire between the transmitter and the receiver to enable them to “share” a clock (as required by certain other protocols). The receiver/transmitter part of the acronym means just what it says: the protocol tells you what signals you need to send from the transmitter and what signals you should expect to acquire at the receiver.

The first term of the acronym, “universal,” is a bit more puzzling. According to *Wikipedia*, the term “universal” refers to the fact that the data format and the speed of transmission are variable. My feeling has always been that the

term “universal” is basically hype; someone probably figured a “universal asynchronous receiver/transmitter” would sell better than a simple “asynchronous receiver/transmitter.”

TEAMWORK NEEDED

Before you can use a UART to transfer information from device to device, the transmitter and receiver have to agree on a few things. First, they must agree on a transmission speed. They must agree that each transmitted bit will have a certain (fixed) duration, denoted T_{BIT} . A 1/9,600-s duration is a typical choice, related to a commonly used crystal’s clock speed, but there are many other possibilities. Additionally, the transmitter and receiver have to agree about the number of data bits to be transmitted each time, the number of stop bits to be used, and the flow control (if any).

When I speak of the transmitter and receiver “agreeing” about these points, I mean that the people programming the transmitting and receiving systems must agree to use a certain data rate, for example. There is no “chicken and egg” problem here. You do not need to have an operational UART *before* you can use your UART; you only need a bit of teamwork.

UART TRANSMISSION

Using a UART is considered the simplest way of transmitting information.

Figure 1 shows the form the transmissions must always make. The line along which the signal is transmitted is initially “high.” The transmissions begin with a single start bit during which the line is pulled low (as all UART transmissions must). They have eight data bits (neither more nor less) and a single stop bit (and not one and a half or two stop bits) during which the line is once again held high. (Flow control is not used throughout this article.)

Why must this protocol include start and stop bits? The transmitter and the receiver do not share a common clock, so how does the receiver know when a transmission has begun? It knows by realizing that the wire connecting them is held high while a transmission is not taking place, “watching” the wire connecting them, and waiting for the voltage level to transition from high to low, which it does by watching and waiting for a start bit. When the wire leaves its “rest state” and goes low, the receiver knows that a transmission has begun. The stop bit guarantees that the line returns to its “high” level at the end of each transmission.

Transmissions have a start and a stop bit, so the UART knows how to read the two words even if one transmits that data word 1111111 and follows it with 1111111. Because of the start and stop bits, when the UART is “looking at” a line on which a transmission is beginning, it sees an initial low level (the start bit), the high level repeated eight times, a ninth high level (the stop bit), and then the pattern repeats. The start bit’s presence enables the UART to determine what’s happening. If the data word being transmitted were 00000000 followed by 00000000, then the stop bit would save the day.

The type of UART connection I describe in this article only requires three wires. One wire is for transmission, one is for reception, and one connects the two systems’ grounds.

The receiver and transmitter both know that each bit in the transmission takes T_{BIT} seconds. After seeing a voltage drop on the line, the receiver waits for $T_{BIT}/2$ s and re-examines the line. If it is still low, the receiver assumes it is in the middle of the start bit. It waits T_{BIT} seconds and resamples the line. The value it sees is then used to determine data bit 0’s value. The receiver then samples every T_{BIT} seconds until it has sampled all the data bits and the stop bit.

A FORGIVING PROTOCOL

Up until now, the assumption is that the transmitter and the receiver each expect bits to take the same amount of time, exactly T_{BIT} seconds. There are two reasons this is a bad assumption. First, the transmitter and receiver have their own clocks. Even if the clocks are supposed to be 100% precise, they

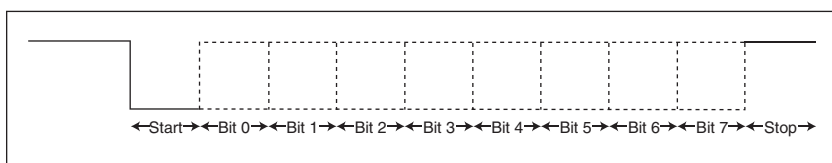


FIGURE 1

The waveform output by a microprocessor’s UART is shown. While “at rest,” the UART’s output is in the high state. The transmission begins with a start bit in which the UART’s output is low. The start bit is followed by eight data bits. Finally, there is a stop bit in which the UART’s output is high.

won’t be. Second, sometimes it can be hard or impossible to make T_{BIT} the same at both the transmitter and the receiver. What happens when there is a mismatch?

Let’s call the bit duration at the transmitter T_{BIT} and the bit duration at the receiver $T_{BIT} + \Delta$. Following the logic shown in **Figure 2**, the receiver samples the waveform it sees at the times:

$$(T_{BIT} + \Delta) 0.5, (T_{BIT} + \Delta) 1.5, \dots, (T_{BIT} + \Delta) 8.5, (T_{BIT} + \Delta) 9.5$$

As long as:

$$|9.5\Delta| < 0.5 T_{BIT}$$

all of the examination times occur in the correct bit, and no real harm has been done. The rule of thumb is that as long as:

$$\frac{|\Delta|}{T_{BIT}} < 0.05$$

and the rates are within about 5% of one another, the UART will work well. This is, indeed, a “forgiving” protocol.

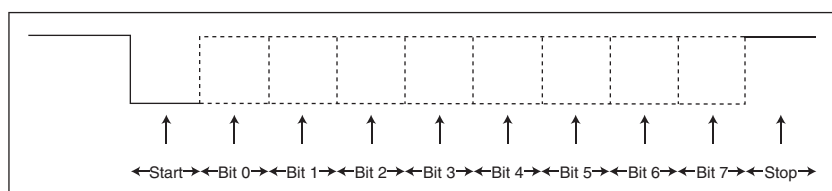
CONNECT THE UART TO A 5-V DEVICE

Though a microprocessor’s I/O pins generally carry logic-level signals, most devices that communicate with one another via a UART expect the communications to be carried out using the higher voltage levels set in the RS-232 standard. RS-232 requires that the signal’s voltages be between 3 and 15 V and between -3 and -15 V. The -3-to-3-V range is forbidden. (A Logic 1 in the data signal corresponds to a negative voltage and a Logic 0 corresponds to a positive voltage.)

As a rule, when connecting a microcontroller to another piece of equipment, you use a chip (e.g., Maxim Integrated’s MAX232) to perform the level shifting. The microprocessor outputs voltages at its logic levels (e.g., 0 and 5 V) to the level shifter. The level shifter ensures that

FIGURE 2

This is the waveform with the examination times. The receiver must detect the line moving from its “rest position,” from the higher voltage level, to the lower voltage level. Half a period after detecting this change, the receiver resamples the line to see if the level is still low. If it is, the receiver samples nine more times. It samples eight times in what should be the middle of each of the data bits and then it samples what should be the middle of the stop bit.



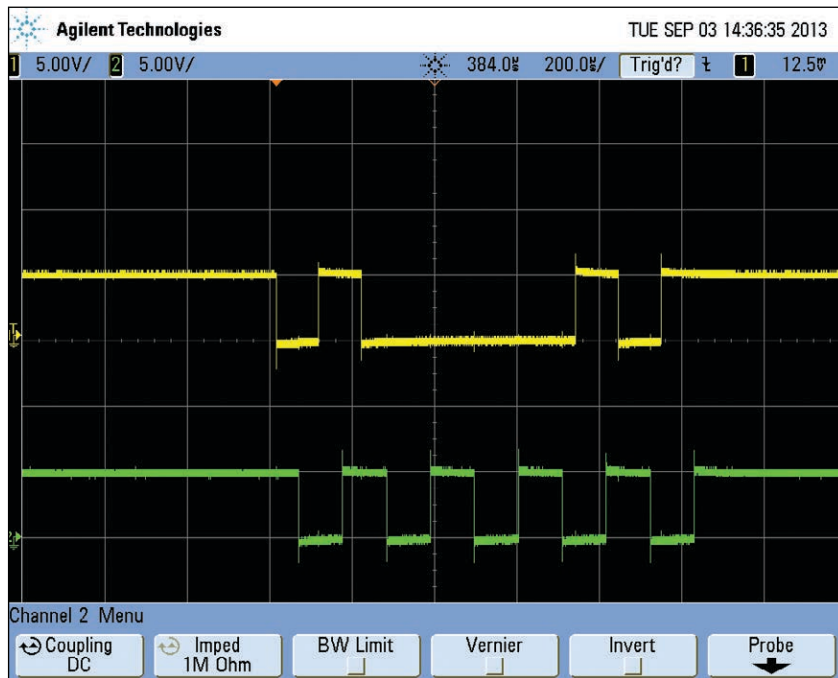


PHOTO 1

The received signal (in yellow) and the times at which timer interrupts occurred (indicated by a change in the state of the green trace) are shown. Because the timer interrupt does not occur with the exact desired frequency, the sampling times do not always occur in the middle of the bit being sampled.

the wires connecting the devices operate using the higher voltages required by the RS-232 standard.

For the projects described in this article, I used the evaluation kit for Analog Devices's ADuC841. The ADuC841's UART output is a signal that is either 0 or 5 V. Analog Devices includes a special communications cable with the evaluation kit that has a level-shifting chip inside. You connect the cable to the ADuC841 on one side and the computer's COM port on the other. The cable's level-shifting chip handles the level shifting.

USING A UART

When using an 8051-type microcontroller to transmit data with a UART, you write to an 8-bit register and the UART takes the 8 bits and creates an electrical signal (see **Figure 1**). The

UART creates the signal by prepending a start bit, sending the data bits least significant bit (LSB) first, and, finally, appending a stop bit. In such microcontrollers, when the UART finishes transmitting a byte, it asserts a "transmit interrupt." The programmer causes the UART to transmit a multicharacter string by having it transmit a new character and a new byte each time the transmit interrupt is asserted. This continues until the end of the string is reached. At that point, the programmer ignores the transmit interrupt and the UART patiently waits until it is needed again.

"BUILDING" A TRANSMITTER

Let's see how a microcontroller, in this case the ADuC841, can use one timer and one other "spare" interrupt to efficiently transmit a string. A spare interrupt is co-opted, in this case external interrupt 0, and used like the 8051-standard UART's transmit interrupt. A timer, in this case Timer1, takes care of all the timing issues associated with transmitting one bit every T_{BIT} seconds. Let's consider how the timer is used.

In the program I wrote, I was aiming for a 9,600-bps transmission rate, so I set the timer to interrupt about every 104 μ s. (Since the UART protocol is forgiving, there was no need to achieve 100% accuracy.) Each time a Timer1 interrupt occurred, the microprocessor checked to see which bit needed to be transmitted next and transmitted that bit. When it finished transmitting all the bits, it asserted the external interrupt 1 interrupt.

How does the Timer1 interrupt subroutine know what the relevant bit is? The Timer1 interrupt subroutine forces an external interrupt 1 interrupt each time it finishes transmitting the last bit involved in a data byte transmission, which is the stop bit. The external interrupt 1 interrupt subroutine checks to see the next byte to be transmitted, reads the byte, and translates it into an array of ones and zeros (adding the start and stop bits to the array). It stores this array in the data segment. When the Timer1 interrupt needs a bit to transmit, it looks at this array. When the external interrupt 1 interrupt subroutine comes to the end of the array to be transmitted, it does not do any processing and returns. To initiate a transmission, you position a pointer at the beginning of the array to be output and force an external interrupt 1 interrupt.

Once everything is programmed, you can transmit strings at 9,600 bps as usual. However, you do not need a built-in UART.

DESIGNING A RECEIVER

To understand a protocol, you don't implement half of it; you implement all of it.



circuitcellar.com/ccmaterials

Wikipedia, "RS-232."

—, "Universal asynchronous receiver/transmitter."

SOURCES

ADuC841 Microcontroller

Analog Devices, Inc. | www.analog.com

MAX232 Dual driver/receiver

Maxim Integrated, Inc. | www.maximintegrated.com

RESOURCES

S. Engelberg, *ADuC841 Microcontroller Design Manual: From Microcontroller Theory to Design Projects*, Circuit Cellar, Inc., 2011.

ABOUT THE AUTHOR

Shlomo Engelberg (shlomoe@jct.ac.il) received his BE and ME degrees in engineering from The Cooper Union and his PhD in mathematics from New York University's Courant Institute. He is an associate professor in the electronics department of the Jerusalem College of Technology. From December 2008 until June 2012, Shlomo was the editor-in-chief of the *IEEE Instrumentation and Measurement Magazine*. He is the author of many articles and several books, the latest of which are *Digital Signal Processing: An Experimental Approach* (Springer, 2008) and *ADuC841 Microcontroller Design Manual: From Microcontroller Theory to Design Projects* (Circuit Cellar, 2011). His research interests include applied mathematics, instrumentation and measurement, signal processing, and control theory, and he has a soft spot for 8051-based microcontrollers.

In addition to the simple transmitter, I implemented a simple receiver. I utilized both of the ADuC841's external interrupts. I used one as an edge-triggered interrupt and one because I needed to "repurpose" some interrupt, and it was free. I used Timer0 to handle all the timing issues.

I connected the input from the (level shifter connected to the) computer to the pin associated with the edge-triggered interrupt, P3.2. The edge-triggered interrupt on 8051-type microcontrollers is triggered by a falling edge. Since a falling edge indicates the start of a word, that is just the kind of interrupt I needed. After detecting a falling edge, the microcontroller set up the timer to interrupt after half a bit period (about 52 μ s). After half a bit period, when the timer interrupt occurred, the microprocessor checked that the input to P3.2 was still low. If P3.2 was no longer low, the microprocessor decided that no transmission had been detected and readied itself to receive the next "real" transmission. If P3.2 was still low, then the microprocessor set the timer to interrupt in another bit period (about 104 μ s).

At the next timer interrupt, the value on P3.2 (i.e., the value input) was shifted into the accumulator and the timer was set to interrupt in another bit period. This went on until all the data was shifted to the accumulator.


After one more bit period, the microprocessor checked to ensure there was a stop bit. If there was, then the microprocessor stored the received value and forced external interrupt 1, which I used as the "receive interrupt." To ensure everything was working correctly, I had the "receive interrupt" check to see if the value received was 65 (i.e., was the ASCII code of "A"). If it was, then I had the microprocessor toggle an LED state. The program works like a charm.

A BIT ABOUT DEBUGGING

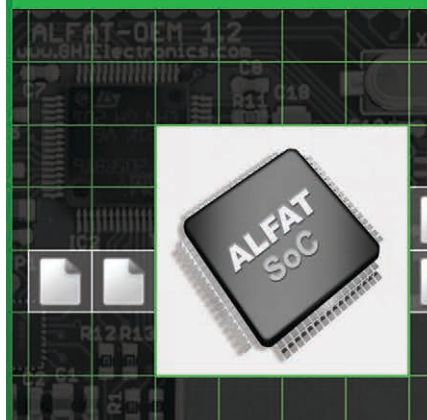
I wrote the receiver program at home while working with an IDE but without an actual microcontroller. When I finished using the IDE to debug with the microcontroller, everything seemed to work fine. The next day I started debugging the program while it was running on an ADuC841 and nothing worked right. It turned out that I had forgotten to reload the timer in one place and that messed up the timing. As part of my debugging effort, I had an I/O pin change state each time the program entered the timer interrupt subroutine. Comparing that with the computer's input should have shown me state changes on the I/O pin in the middle of each received bit. That is not what I saw and that is (part of) how I realized my mistake.

Even after fixing the problem, I left the extra lines of code. **Photo 1** shows the signal being input (in yellow) and the bit that is supposed to toggle in the middle of each signal bit. Because of the way I set up the timer, the period is not exactly correct. You can see that toward the end of the input signal the changes of state occur near the "far end" of the relevant bit. Because the protocol is so forgiving, there was no urgent need to readjust the timer's timing. The program works nicely as it is.

TRY IT YOURSELF

The transmitter and the receiver are both fairly simple to write. I enjoyed writing them. If you like playing with microprocessors and understanding the protocols with which they work, you will probably enjoy writing a transmitter and receiver too. If you do not have time to write the code yourself but you'd like to examine it, feel free to e-mail me at shlomoe@jct.ac.il. I'll be happy to e-mail the code to you. 

File System Solutions



With **ALFAT SoC** You Can

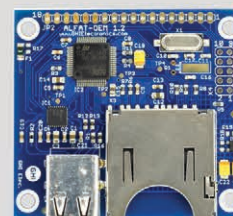
Use FAT/FAT32 with Long File Name

Use UART, SPI or I2C serial interface to access files.

Use SD-Reader Mode allowing the ALFAT SoC to act as an SD Card Reader.

Access files on two USB drives and an SD card simultaneously

Use high speed USB and 4-bit SD card interfaces.



The ALFAT SoC is also available as an OEM board.

GHIElectronics.com



DesignSpark Tips & Tricks

Day #11: Working with Large Designs

By **Neil Gruending**
(Canada)

Today let's try a larger, more complex design in DesignSpark PCB that uses multiple schematic sheets.

So far I've been using small design examples that fit on a single schematic sheet. Today let's try using multiple schematic sheets in DesignSpark PCB as you would for a larger design.

Creating multipage schematics

I usually like to partition a larger design into multiple schematic pages to help break it up into multiple logical blocks. Each page has its own unique net list, and connections between sheets are done

with "ports" or "sheet connectors" depending on the PCB package. DesignSpark PCB can do all of this with a little effort.

Once a schematic has been added to a project file, DesignSpark will make sure that it has unique reference designators across the entire design and will warn you if it finds any duplicates. Any added components will also get unique designators. Net connections on individual pages are also assigned unique names. But you can create net connections between sheets by giving it a name, which makes it global across the project. The name can be anything, but I recommend that it doesn't start with "N" followed by numbers, which is the default DesignSpark naming convention. DesignSpark already includes power and ground components that automatically create global power nets. Although you can use custom net names to join nets across pages, I find it makes a schematic very difficult to follow and understand. So instead we'll create "port" symbols like those in **Figure 1**, which shows an example of an input port, bidirectional port, and an output port. They are just a simple one-pin component with an arrow to indicate the signal direction.

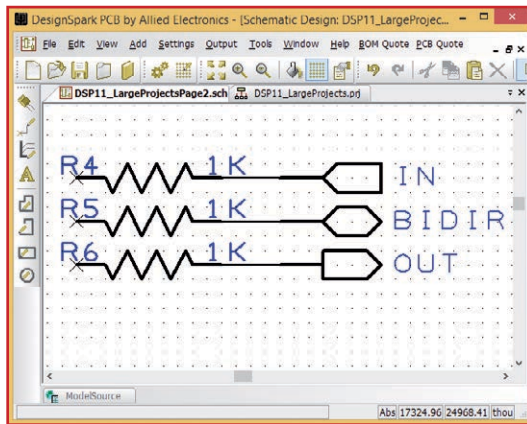


Figure 1.
Schematic port symbol example.

You use the ports by placing them on a schematic page and wiring them appropriately. DesignSpark doesn't allow library components to show net names automatically at a fixed position so you will need to do that manually. When you right click on a wire, you can select the "Display Net Name" to show the net name and then manually move it how you would like it. Don't forget to change the net name to a custom one by right clicking on the net name and selecting the "Change Net" option because

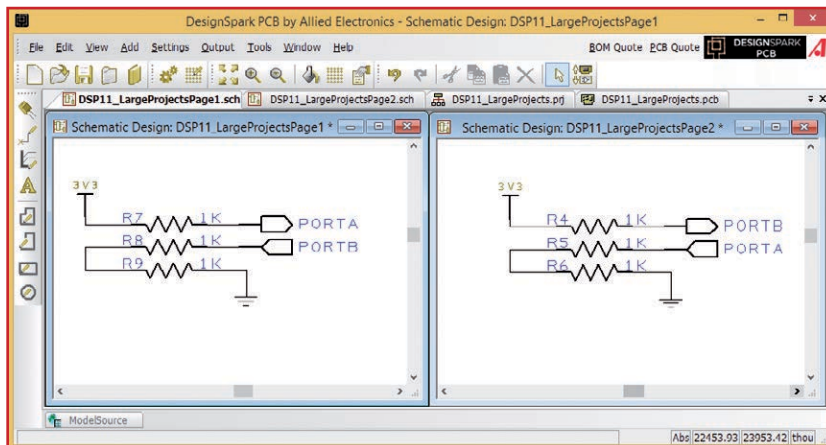


Figure 2.
Multipage schematic example.

DesignSpark will automatically hide all of the default net names in the “Change Net” window, making them impossible to select.

Let’s use the two-page schematic in **Figure 2** for the rest of our discussion. I’ve kept it simple, so it’s easy to see how net connectivity works in DesignSpark.

Creating the PCB

Now let’s import our design into a PCB to make sure that the PORTA and PORTB nets are connected properly. First, you create the PCB file using the File->New command and checking the “Add to Open Project” box, which will give you a blank PCB. Now open one of the schematic pages and use the “Tools->Forward Design Changes” menu to add the components and nets to the PCB. After a little editing, I got a PCB like the one in **Figure 3**. I labeled all of the net connections to show that the netlist was properly imported, including the PORTA and PORTB nets.

Cross-probing nets

Our example multiple page schematic is pretty simple, so it’s easy to remember the entire schematic while working on it. But this becomes an issue for large, complex designs when you are trying to understand a net connection and you want to do something like match a net connection in a PCB to a schematic. This is called cross-probing the nets and it’s one of the new features in DesignSpark PCB version 6.0.

So how do you cross-probe a net in the new version? Use the “Edit->Cross Probe” menu to go into cross-probe mode. Now, every net connection you click on a net in the PCB will automatically be selected and shown in the schematic. If you click on a net in the schematic, then it will be selected and shown in the PCB as well. Once you’re done probing nets the “Edit->Cross Probe” menu will exit cross-probe mode.

The PORTA and PORTB sheet-to-sheet connections in our example

project are a special case for cross-probing. When you select one of those nets in the PCB, DesignSpark will highlight all of the PORTA/B connections but it will only open one of the schematic pages for viewing. The easy way to see all of the PORTA/B connections is to go into cross-probe mode in the schematic sheet and select the net. This makes DesignSpark open the rest of the schematic pages that contain that net and display them all simultaneously. For example, **Figure 4** shows how DesignSpark highlights the PORTB net when I clicked on it from the PCB with this procedure.

Web Link

- [1] www.rs-online.com/designspark/electronics/eng/knowledge-item/designspark-pcb-cross-probe-overview

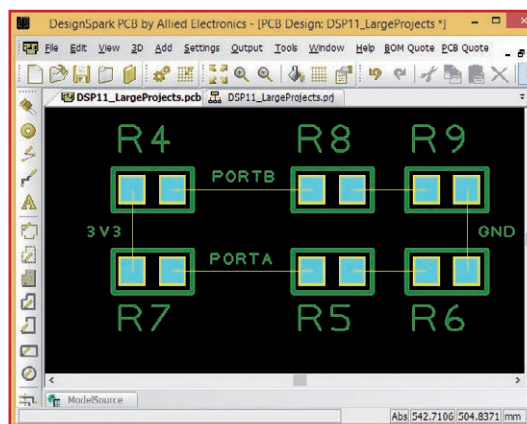


Figure 3. Imported PCB.

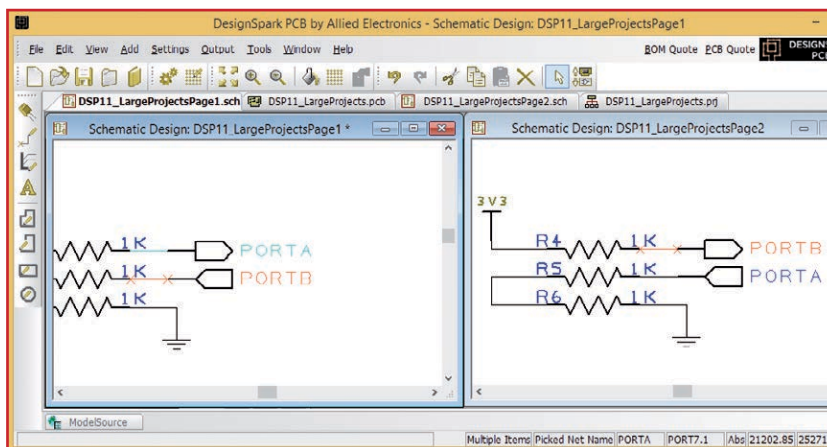


Figure 4. Cross-probing nets.



Varactor Diodes

Weird Component #6

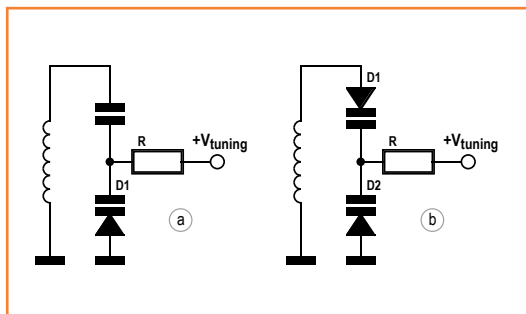
By **Neil Gruending**
(Canada)

If you've ever touched any RF circuitry, there's a very good chance it contained at least one variable capacitor in its voltage controlled oscillators (VCOs) and filter circuits. They used to use mechanical variable capacitors. But varactor diodes, also known as varicaps, are now preferred because of their small size, high performance, and excellent tracking ability. **Figure 1** shows the schematic symbol for them.

Figure 1.
Varactor (varicap) schematic symbol.



Figure 2.
Varactor circuit examples.



All semiconductor PN diode junctions have a small amount of variable capacitance when they're reversed biased. The reverse bias on the diode PN junction creates an insulator region at the junction that creates a small parallel plate capacitor. The insulator width will increase with the bias voltage which decreases the effective capacitance. This is what gives diodes their variable capacitance. Varactor diodes, however, are specifically optimized for their capacitance properties and have two distinct types based on their adjustable range. Abrupt varactor diodes change a small amount of capacitance for a change in bias voltage which gives them high Q (quality factor) and low distortion properties. Hyperabrupt diodes, on the other hand, have a much larger change in capacitance over the same bias voltage change, which can increase distortion. Abrupt diodes can have a capacitance range of 2:1 whereas hyperabrupt diodes typically have a capacitance range of 10:1. In the past, power varactors of the hyperabrupt variety were used as frequency

multipliers (mostly triplers), providing a cheap way for radio hams to produce a few watts of RF power on UHF such as 70 cms (144 MHz \times 3) or SHF such as 23 cms (430 MHz \times 3). The method is suitable for non-linear modes only (e.g., FM and CW).

Figure 2 shows two varactor diode circuit examples. Both circuits are using the varactor diode to adjust an LC tank circuit by changing the diode reverse bias voltage V_{tuning} . Figure 2a is a simpler design and works well with low amplitude signals. However, when the signal gets too large it will affect the diode bias voltage and cause distortion. Figure 2b uses two diodes to counteract the distortion at higher signal levels. This is because as the signal causes one diode to decrease its capacitance, the other one will increase its capacitance by the same amount. The downside is that the two diodes in series will have half the capacitance, which is significant when we're talking about capacitors in the picofarad range.

The varactor diodes don't need a lot of bias current, so the value for R isn't critical as long as it provides enough isolation between the bias voltage and the tank circuit. Sometimes you will see an inductor used instead, but its function is the same. It's also important to keep the diodes reverse-biased because they will create a lot of spurious noise if they start to forward conduct. If you want to experiment with varactor diodes but don't have any, try a regular diode from your parts box. Lots of people have had success using 1N400x diodes, power rectifier diodes, zener diodes and even LEDs. Just don't give up—it may take several tries to find a diode that works!

Save \$25 off

CC Vault

The incredible issues archive from your friends at *Circuit Cellar*.



This pocket-sized vault comes fully loaded with every issue of *Circuit Cellar* magazine and serves as an unparalleled resource for embedded hardware and software design tips, schematics, and source code.

From green energy design to 'Net-enabled devices, maximizing power to minimizing footprint, CC Vault* contains all the trade secrets you need to become a better, more educated electronics engineer.

BONUS! Build your archive by downloading your latest-issue PDFs straight to the drive! Personalize your CC Vault by adding *Elektor* or *audioXpress* issue archives, available as an add-on during time of purchase, or your very own project files.



*CC Vault is a 16-GB USB drive.

Save \$25. Enter coupon: VAULT14 cc-webshop.com

The Sun Chaser

GPS Reference Station Design

The Sun Chaser is a well-designed, energy harvesting system that automatically recalculates a solar panel's position to ensure it is always facing the sun. The project won third prize in the 2012 Renesas RL78 Green Energy Challenge.

By Sjoerd Brandsma (The Netherlands)



When Circuit Cellar and Renesas Electronics announced the 2012 Renesas RL78 Green Energy Challenge, I was excited to participate. I decided to create something fun and educational. My project, which I call the Sun Chaser, was based on my professional background working at a company where using GPS is very important.

The Sun Chaser's solar panel is mounted on a rotating disc. A registered GPS calculates the solar panel's orientation. You can use an external compass, the internal accelerometer, a DC motor, and a stepper motor to determine the solar panel's exact position. The Sun Chaser uses Renesas Electronics's RDKRL78G13 evaluation board running the Micrium μ C/OS-III real-time kernel.

GPS REFERENCE STATION

Whenever you want to know where you are, you can use a GPS receiver that provides your position. A single GPS receiver can provide about 10 to 15 m (i.e., 33' to 50') position accuracy. While this is sufficient for many people, some applications require positioning with significantly higher accuracy. In fact, GPS can readily produce positions that are accurate to 1 m (3'), 0.5 m (18"), or even 1 to 2 cm (less than 1"). A technique called

"differential GPS" can be used to achieve higher accuracy.

The differential technique requires one GPS receiver to be located at a known position (often called a control or reference point) and a second "rover" receiver at the location to be measured. The information from the two GPS receivers (rover and control) is combined to determine the rover's position. That's where a GPS reference station comes in. It functions as the control point and serves potentially unlimited users and applications. Leica Geosystems has published an excellent introductory guide about GPS reference stations (Refer to the Resources at the end of this article.)

The GPS reference station should always be located at a position with a broad sight. In some situations it can be difficult to provide a decent power supply to the system. When regular power isn't available, a solar panel can power the GPS reference station.

My Sun Chaser GPS reference station uses a 10-W solar panel connected to a 12-V battery to provide enough power. To increase the energy harvesting, the solar panel is mounted on a rotating disc that can be controlled by a DC motor to point in the desired direction. A stepper motor controls the solar panel's "tilting."

Photo 1 highlights the main components. This article mainly focuses on the fun part, which is using basic, free, easily obtainable components to build such a system.

USING THE EVALUATION BOARD

The RDKRL78G13 is an evaluation and demonstration tool for Renesas Electronics’s RL78 low-power microcontrollers. A set of human-machine interfaces (HMIs) is tightly integrated with the RL78’s features. I used several of these interfaces to control other devices, read sensors, or store data.

Most of the system’s hardware is related to placing the solar panel in the correct position. **Figure 1** shows the top-level components used to store the GPS information and position the solar panel.

The RDKRL78G13 evaluation board has an on-board temperature and light sensor. Both sensor values are stored on the SD card. The on-board light sensor is used to determine if rotating/tilting makes sense (at night it’s better to sleep). For this project, the temperature values are stored just for fun so I could make some graphs or do some weather analysis.

A micro-SD memory card slot on the RDKRL78G13 evaluation board provides file system data storage. I used it to store all incoming data and log messages using the FAT16/FAT32 file system.

The on-board Renesas Electronics RQK0609CQDQS MOSFET controls the DC motor that rotates the evaluation board (see **Figure 2**). The DC motor can be controlled by applying a PWM signal generated from one of the RL78’s timers. The MOSFET is controlled by the RL78’s T005 port and powered from the 12-V battery. A PWM signal is generated on T005 by using Timer4 as a master and Timer5 as a slave. It’s only necessary to rotate clockwise, so additional hardware to rotate the platform counterclockwise is not required.

A digital compass is needed to determine the evaluation board’s rotated position or heading (see **Figure 3**). The Honeywell HMC5883L is a widely used and low-cost compass. This I²C-based compass has three-axis magnetoresistive sensors and a 12-bit ADC on board. It can read out values at a 160-Hz rate, which is more than enough for this project.

The compass uses the RL78’s IICA0 port through the Total Phase Beagle debug header, which is mounted on the RDKRL78G13 evaluation board. The Beagle analyzer provides easy access to this I²C port, which increases the flexibility to change things during prototyping.

The HMC5883L compass turned out to be

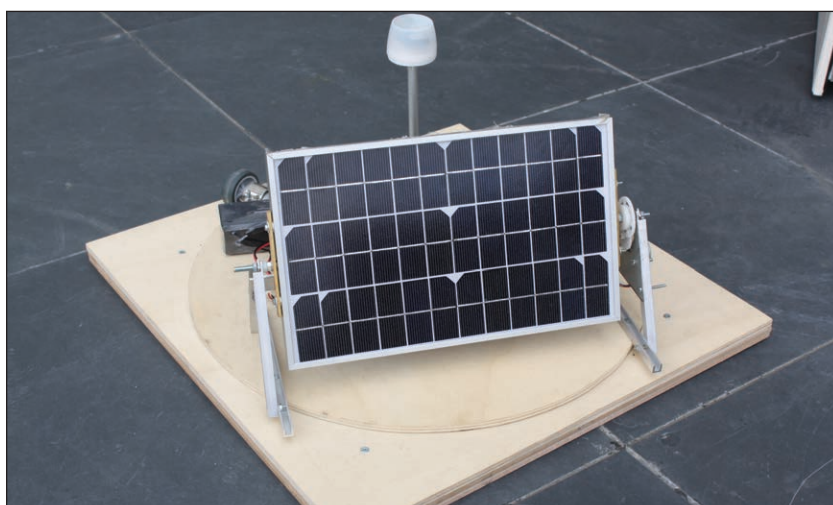


PHOTO 1
The Sun Chaser’s stepper motor controls the solar panel’s “tilting.”

a very sensitive device. Even the slightest change in the hardware setup seemed to influence the results when rotating. This meant some sort of calibration was needed to ensure the output was consistent every time the system started.

Later in this article I describe how the HMC5883L can be calibrated. It’s important to know that every time the system starts, it makes a full turn to calibrate the compass.

A GPS module must be connected to the system to provide the system’s current location. I wanted the GPS module to be inexpensive, 3.3-V based, and have an easy and accessible interface (e.g., UART).

Figure 4 shows a schematic of a Skylab M&C Technology SKM53 GPS module, which is based on the MediaTek 3329 GPS receiver module. This module supports NMEA messages and the MTK NMEA Packet Protocol interface

FIGURE 1
The Sun Chaser’s components include a Renesas Electronics RDKRL78G13 evaluation board, a GPS receiver, a stepper motor, and an SD card.

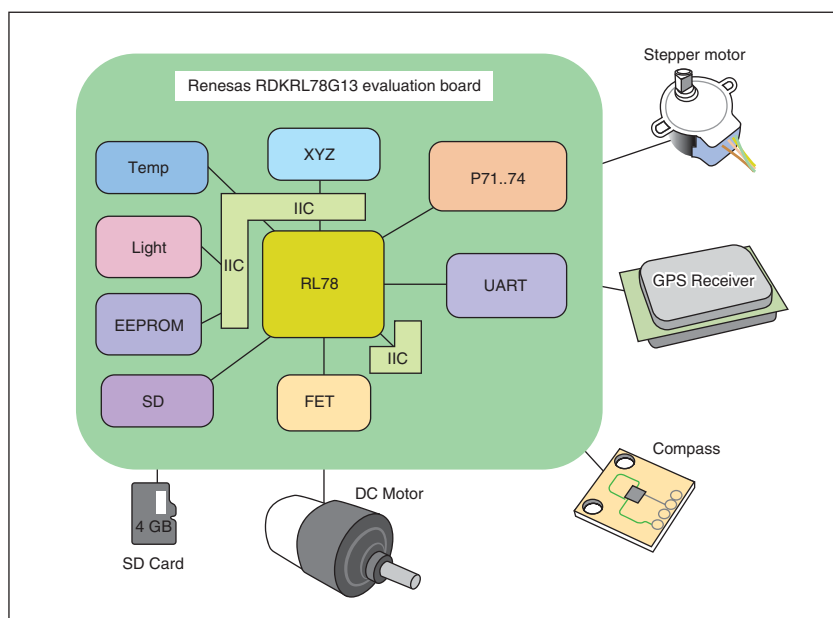


FIGURE 2

A Renesas Electronics RQK0609CQDQS MOSFET helps rotate the evaluation board.

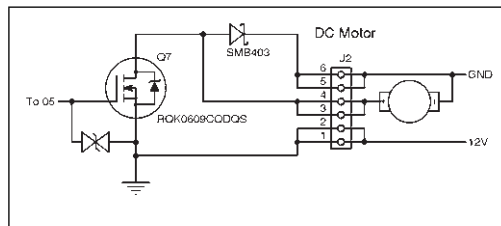


FIGURE 3

A Honeywell HMC5883L digital compass verifies the evaluation board's rotated position or heading.

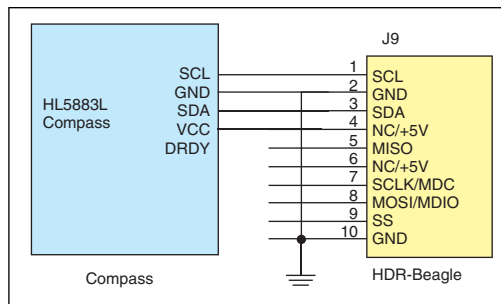
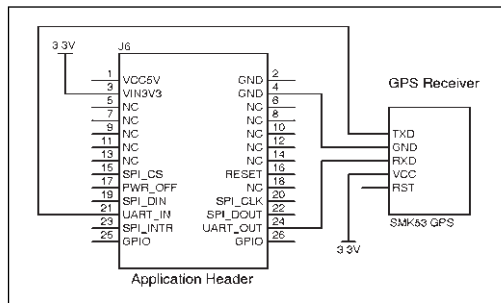


FIGURE 4

A Skylab M&C Technology SKM53 GPS receiver obtains the system's current location.



to control things such as power saving, output message frequency, and differential global positioning system (DGPS).

Unfortunately, the 3329 receiver can't output "raw" GPS data (e.g., pseudorange, integrated carrier phase, Doppler shift, and satellite ephemeris), which would significantly improve the GPS reference station's capabilities. Due to budget and time limitations (it takes some more software development effort to handle this raw data), I didn't use a receiver that could output raw GPS data.

The SKM53 GPS receiver is connected to the RL78's UART2. All data from the GPS receiver is stored on the SD card. As soon as a valid GPS position is received, the system

calculates the sun's position and moves the platform into the most ideal position.

A compact stepper motor is needed to tilt the platform in very small steps. The platform had to be tilted from fully vertical to fully horizontal in approximately 6 h when the sun was exactly following the equator, so speed wasn't really an issue. I wanted to do very fine tilting, so I also needed a set of gears to slow down the platform tilting.

I used an inexpensive, easy-to-use, generic 5-V 28BYJ-48 stepper motor (see **Figure 5**). According to the specifications, the 28BYJ-48 stepper motor has a 1/64 gear reduction ratio and 64 steps per rotation (5.625°/step) of its internal motor shaft.

An important consideration here is that you don't want to retain power on the stepper motor to keep it in position. This particular stepper motor has some internal gears that prevent the platform from flipping back when the stepper motor is not powered.

The stepper motor can be controlled by the well-known ULN2003 high-voltage high-current Darlington transistor array. The ULN2003 is connected to P71-P74. Each of the ULN2003's four outputs is connected to one of the stepper motor's coils. When two neighbor coils are set high (e.g., P72 and P73), the stepper motor will step in that direction.

When it comes to solar panels, you can build your own panel out of individual solar cells or buy a fully assembled one with known specifications. I used a no-name 10-W solar panel. The size (337 mm long × 205 mm wide × 18 mm high) was acceptable and it delivered more than enough energy. I used a charge controller to protect the battery from overcharging and to prevent it from supplying power to the solar panel at night.

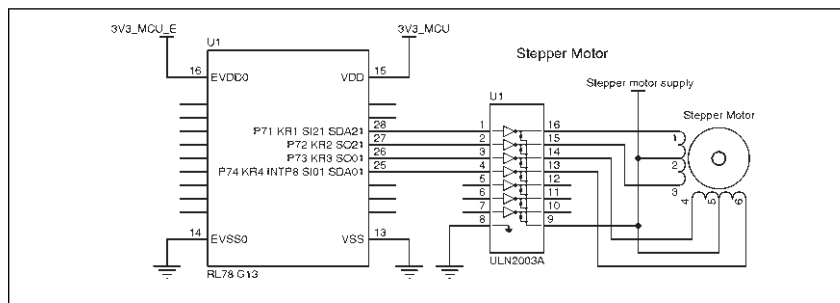
Like solar panels, many charge controllers and battery protectors can be used in such a system. I chose the lazy approach: Just take one off the shelf. The CMP12/24 charge controller is specially designed for small solar systems. It has a stabilized 12-V output, which is taken from the connected battery. It can handle up to 12 A of charging or load current and, according to the specifications, it consumes about 20 mA of quiescent current. There is some room for improvement, but it worked for my project.

I had some 7805 voltage regulators lying around, which I figured could do the job and supply just enough power when the system was starting up. However, when it comes to power saving, the 7805 is not the way to go. It's a linear regulator that works by taking the difference between the input and output voltages and burning it up as wasted heat.

What I needed was a switching regulator or a buck converter. I used a National

FIGURE 5

A 28BYJ-48 stepper motor tilts the platform.



Great Products Great Prices • Great Support

USB-1608G

16-Channel, 16-bit DAQ



Average Rating: ★★★★★ 4.8 out of 5

"Easy to use. Good support." – NoxSC

"Great value for the money." – Frank260

Only \$399

USB-201

8-Channel, 12-Bit DAQ



Average Rating: ★★★★★ 4.8 out of 5

"Great product, great price point." – LabGuy

"Great value and ease of use." – David52

Only \$99

USB-TC

8-Channel, 24-Bit Temperature



Average Rating: ★★★★★ 4.7 out of 5

"Easy, accurate, reliable." – Patrick

"USB-TC is bullet proof." – Greg

Only \$359

USB-500 Series

Stand-Alone Data Loggers



Average Rating: ★★★★★ 4.6 out of 5

"Easy-to-use, understandable results." – Ltmmom

"Out-of-the-box great value!" – DrTom53

From \$49

mccdaq.com

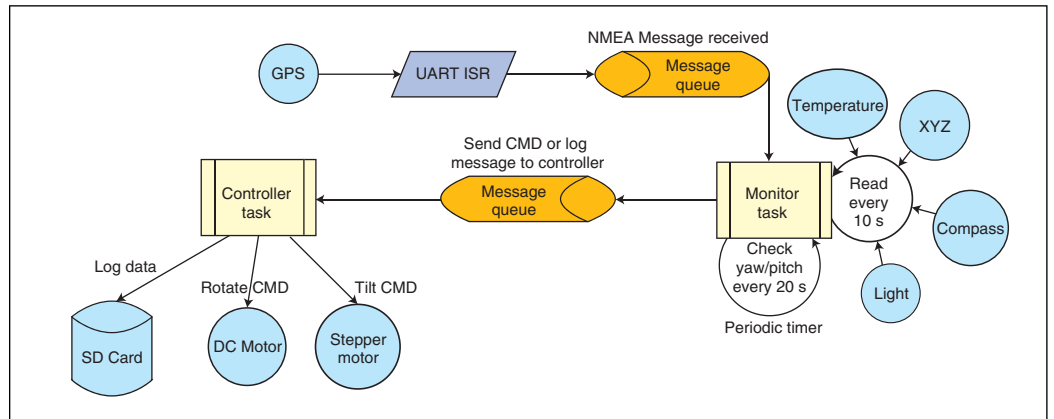


The Value Leader in Data Acquisition

Contact us
1.800.234.4232

FIGURE 6

The Sun Chaser's monitor collects data and instructs the controller. The controller stores data or moves the platform.



Semiconductor (now Texas Instruments) LM2596. Note: The LM2596 is made by several companies and is available in inexpensive, high-quality modules (most cost a little more than \$1 per converter). These ready-to-use modules already have the necessary capacitors, diodes, and so forth on board, so it's really a matter of plug and play.

I used a lead acid RT1219 12-V 1.9-AH battery for power storage. You can use any 12-V battery with sufficient capacity.

MONITOR AND CONTROLLER

The system runs on the $\mu\text{C}/\text{OS-III}$ real-time kernel. For this purpose that may be overkill, but it's always good to explore new things.

From a software point of view, the system can be divided into two parts: a monitor and a

controller. The monitor collects data and tells the controller what to do. The controller can simply store data to the SD card or tilt/rotate the platform until it's in the desired position (see Figure 6).

When the system is powered up, it waits until a valid NMEA GPRMC message is received. Known as the "recommended minimum" sentence, this is the most common sentence transmitted by GPS devices. This one sentence contains nearly everything a GPS application needs: latitude, longitude, speed, bearing, satellite-derived time, fix status, and magnetic variation. The monitor can use this information to calculate the sun's position and the solar panel's ideal position.

When the solar panel's ideal position is calculated and a platform adjustment is needed, the monitor task uses a message queue to send a message to the controller task. The controller task waits for messages from the message queue and starts doing "work" when a message is received. When an adjustment command is received, it will control the DC motor or stepper motor to adjust the solar panel. If a log command is received, the controller will use the FAT16/32 file system to store the log data to the SD card.

A $\mu\text{C}/\text{OS-III}$ periodic timer with a 20-s interval is used to check if the platform needs adjustment. When this timer expires, the ideal position is newly calculated based on the current time and position. When adjustments are needed, the monitor task sends new adjustment commands to the controller task. The system uses another periodic timer to read all the sensors. When this timer expires, the temperature, accelerometer values (XYZ), compass, and light sensor are read and sent to the controller task as a log message.

Listing 1 shows a typical log message. In the listing, T is temperature, L is light, C is compass heading in degrees, and X and Y are raw values. The last two columns X and Y are read from the accelerometer.

LISTING 1

This log message includes information about the temperature, light, and compass heading.

```
2012 8 9 15:36:49 |T|29.016|L|1278|C|245|-9|91|X|51|Y|-7
2012 8 9 15:37: 9 |T|29.078|L|1474|C|246|-10|89|X|52|Y|-4
2012 8 9 15:37:29 |T|29.078|L|1545|C|246|-10|92|X|51|Y|-6
2012 8 9 15:37:49 |T|29.078|L|1610|C|246|-10|93|X|52|Y|-5
2012 8 9 15:37: 9 |T|29.016|L|1668|C|245|-9|88|X|52|Y|-6
```

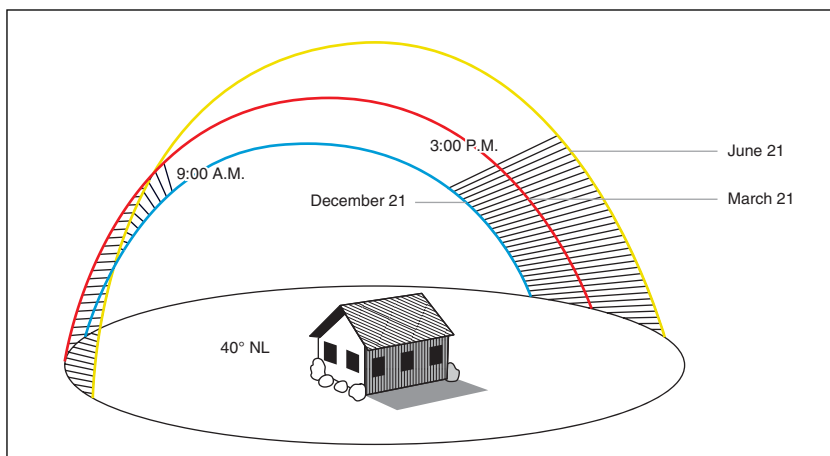


FIGURE 7

The sun's morning and the afternoon positions vary during the winter, spring, and summer months.

SOLAR PANEL ORIENTATION

One of the system's core functions is to orient the solar panel to the sun as accurately as possible. The sun's position can be calculated if you know your location and time of the day. **Figure 7** shows where the sun is during the day for several seasons of the year.

The system relies on an accurate position determined by the GPS receiver and an algorithm to calculate the sun's location based on this position and the current time. These algorithms (and code) are freely available on the Internet. For the Sun Chaser project, I used the C# code in C. Cornwall, A. Horiuchi, and C. Lehman's 2014 document, "Sunrise/Sunset Calculator" to make a port to C. The C# code was created for the US Department of Commerce, National Oceanic & Atmospheric Administration (NOAA) Earth System Research Laboratory (ESRL).

TESTING AND DEBUGGING

One of the things I ran into when debugging the system was the following. The HMC5883L seems to be an easy-to-use compass. You simply hook it up to an I2C port and read the compass value. At least, that's what I thought. It turned out to be a bit more difficult because I discovered that the compass did not provide very consistent results. Different results were produced based on the compass' placement on the board and the way in which it was placed inside or outside.

Figure 8 shows in blue the uncalibrated values that were returned by the compass. For each step the heading/yaw was registered and logged to the SD card. It's clear that from step 37 to 64 the line's slope is very different from the rest. The orange line would be the ideal situation, having a fixed value for each step.

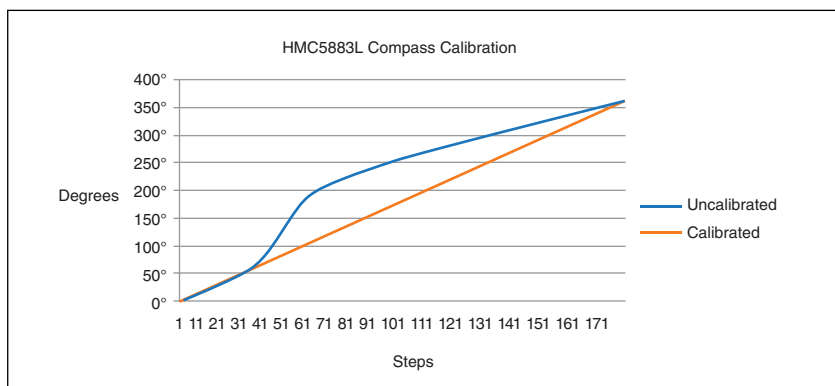
The heading/yaw values are calculated based on the raw X and Y values (see the HMC5883L datasheet for more information) retrieved from the compass. When the platform is fully rotated, the raw X and Y values can be plotted (see **Figure 9**).

The blue points are the raw values directly read from the compass. They clearly have a significant offset to the lower right. To compensate, add an offset to the blue points in such a way that the center of the red circle is on 0,0.

The compensation value can be calculated by averaging the raw X values and the raw Y values:

$$\text{Calibration X} = \frac{(X1 + X2 + X3 + Xn)}{n}$$

$$\text{Calibration Y} = \frac{(Y1 + Y2 + Y3 + Yn)}{n}$$



The calibration values can be applied to the raw values, which results in a clean circle and a linear line when converted to degrees.

The compass calibration can be initiated by keeping Switch 3 pressed when booting the system. It then will make a full round and calculate the calibration values. When these values are calculated, it will store them in the EEPROM and will use them every time the system is restarted.

OBSTACLES AND ISSUES

I based the project's entire hardware design on some rough ideas and the availability of inexpensive and easily obtainable components. The fun part is solving all kinds of problems as soon as you get the components that basically fit into the vague design. One of the challenges was how to rotate the platform without having anything connected to the base plate.

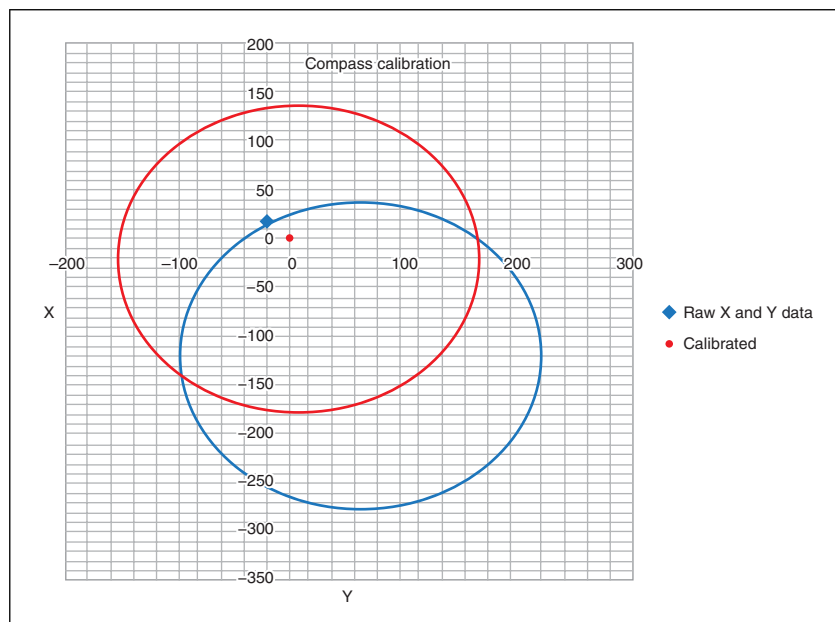
All the electronic devices, including the DC motor responsible for rotating the platform, should be mounted on the same surface where

FIGURE 8

The blue line shows the uncalibrated values returned by the compass. The orange line shows what happens when there is a fixed value for each step.

FIGURE 9

X and Y values are plotted for a full platform rotation.



ABOUT THE AUTHOR

Sjoerd Brandsma (sbrandsma@gmail.com) lives in Kerkwijk, The Netherlands. He received his BSc in Computer Science in 2000 and worked as an embedded software engineer until 2008. Sjoerd is currently an R&D manager at CycloMedia. He enjoys designing with cameras, GPS receivers, and transceivers.

the battery is located. The easiest solution was to place the DC motor on the edge of the rotating plate and use a small wheel with a rubber surface to grip the base plate. The stepper motor I used to tilt the platform also caused some trouble because the gear did not fit on the stepper motor's much larger axle. The stepper motor needed to be taken apart completely to adjust the axle in such a way that the gear fit well.

One of the major software issues was caused by the fact that a MultiMediaCard/Secure Digital (MMC/SD) driver was not


available nor was a FAT16/FAT32 library provided with the RDKRL78G13 evaluation board. Some Renesas Electronics enthusiasts on the RenesasRulz.com forum pointed me in the direction of Electronic Lives Manufacturing's FatFs generic FAT file system module.

"FatFs is a generic FAT file system module for small embedded systems," according to Electronic Lives Manufacturing's website. "The FatFs is written in compliance with ANSI C and completely separated from the disk I/O layer. Therefore it is independent of hardware architecture. It can be incorporated into low-cost microcontrollers, such as [Atmel's] AVR and 8051, [Microchip Technology] PIC, ARM, [Zilog] Z80, 68k."

After inspecting the source code, I found this was an excellent candidate to port to the RDKRL78G13. The MMC/SD driver just needed some minor modifications to correctly communicate with the micro-SD card.

FOLLOWING THE SUN

After all the problems were solved it was time to sit back, relax, and watch the system do its job. Actually, that was quite boring because the platform only rotates 0.5" once every 5 to 10 min. To make it a bit more exciting, I installed a camera mounted on a tripod that took a picture every 3 min. This created a time lapse video that nicely demonstrates the platform's movements (see Resources for a link to my YouTube video).

The Sun Chaser perfectly follows the sun's path and keeps the battery fully charged when there's enough sunlight. It can power a small electronics system as long as there's enough sunlight and no rain, which would damage the system due to lack of protection. This project also demonstrates that it's possible to build an interesting green-energy system with a tight budget and a limited knowledge of electronics. 

Editor's Note: For other projects from the 2012 Renesas RL78 Green Energy Challenge, visit <http://circuitcellar.com/contests/renesas-RL78challenge>.

PROJECT FILES



circuitcellar.com/ccmaterials

A. Zaugg, *The Complete Handbook of Solar Air Heating Systems*, Knowledge Publications, 2009.

SOURCES

HMC5883L Digital compass IC

Honeywell International, Inc. | <http://honeywell.com>

3329 GPS Receiver

MediaTek, Inc. | www.mediatek.com

µC/OS-III Real-time kernel

Micrium, Inc. | <http://micrium.com>

RDKRL78G13 Evaluation board, RL78 micro-controller family, and RQK0609CQDQS MOSFET

Renesas Electronics Corp. | www.renesas.com

SKM53 GPS Module

Skylab M&C Technology Co., Ltd. | www.skylab.com.cn

LM2596 Buck converter

Texas Instruments, Inc. | www.ti.com

Beagle I²C/SPI protocol analyzer

Total Phase, Inc. | www.totalphase.com

RESOURCES

S. Brandsma, "Sun Chaser Time Lapse," YouTube, 2012, www.youtube.com.

C. Cornwall, A. Horiuchi, and C. Lehman, "Sunrise/Sunset Calculator," US Department of Commerce, National Oceanic and Atmospheric Administration (NOAA) Earth System Research Laboratory (ESRL), 2014.

Electronic Lives Manufacturing, "FatFs—Generic FAT File System Module," <http://elm-chan.org>.

Leica Geosystems AG, "GPS Reference Stations and Networks: An Introductory Guide," 2005, www.leica-geosystems.com.

Join the Elektor Community

Take Out a GOLD Membership Now!



Your GOLD Membership Contains:

- 8 Regular editions of Elektor magazine in print and digital
- 2 Jumbo editions of Elektor magazine in print and digital (January/February and July/August double issues)
- Elektor annual DVD-ROM
- A minimum of 10% DISCOUNT on all products in Elektor.STORE
- Direct access to Elektor.LABS; our virtual, online laboratory
- Direct access to Elektor.MAGAZINE; our online archive for members
- Elektor.POST sent to your email account (incl. 25 extra projects per year)
- An Elektor Binder to store these projects
- Exclusive GOLD Membership card



ALSO AVAILABLE:

The all-paperless GREEN Membership, which delivers all products and services, including Elektor magazine, online only.



Take Out Your Membership Now at www.elektor.com/member

Connect with us!



www.facebook.com/elektorim



www.twitter.com/elektor

THE CONSUMMATE ENGINEER

Wireless Data Links (Part 4)

Data Encoding

COLUMNS

The first parts of this article series introduced the topic of wireless communications and discussed radio communications, low-power transmitters and data receivers, noise figures, and data bit recovery. This article explains data encoding and examines multipath distortion and the methods to address it.

By George Novacek (Canada)

Part 3 of this article series focused on receivers and recovery. As the article showed, an asynchronous, non-return-to-zero (NRZ) datastream does not work very well in wireless data communications. It is a basic encoding scheme where the signal follows the digital data and, therefore, does not return to zero after each data bit (see **Figure 1**).

For example, a string of ones will result in a high data level transmitted for its duration, while a string of zeros results in low data level

transmitted for its duration. Such a signal causes the average DC voltage to vary, but a stable average value is instrumental for the data slicer's operation.

In addition, the transmitter and the receiver need to be accurately set to the same baud rate and their clocks must be synchronized. This basic encoding scheme is simple to implement in the transmitter and is used in many hard-wired serial data interfaces (e.g., RS-232), but the timing error, which can be as little as 3%, results in a loss of data integrity.

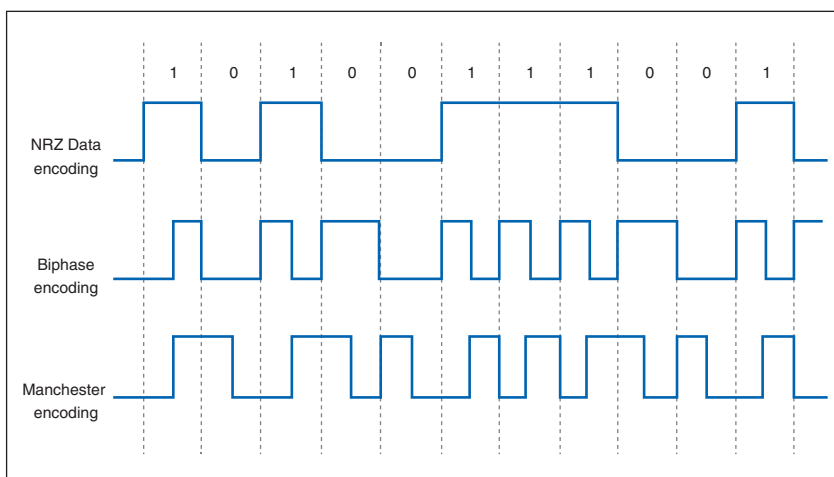
DATA ENCODING TECHNIQUES

One solution to this problem is biphasic encoding, in which a state transition is required at the end of every data bit. In addition, a state transition in the middle of a bit is required when a Logic 1 is being transmitted. With this, the data clock can be synchronized by the bit edges and the data can be recovered.

Manchester encoding is similar to the biphasic scheme. Like biphasic encoding, it enables data rate recovery by the bit edges, and it also yields a zero average DC level, which is needed for reliable data slicer operation. The data slicer and the preceding low-pass filter's time constants are no longer

FIGURE 1

Non-return-to-zero (NRZ) data, biphasic, and Manchester are three types of digital data encoding.



critical. The transmitted RF spectrum and output RF power remain constant, regardless of the data transmitted.

Manchester encoding is achieved by converting every data bit to a sequence of two transmission bits. Data bit 0 is represented by a two-bit sequence 10. Data bit 1 is represented by a two-bit sequence 01 (see **Figure 1**).

The clock synchronization may be augmented by a preamble at the beginning of each data packet. In hardware encoders and decoders, synchronization is usually performed by a phase-locked loop (PLL), while software routines use counter timers.

The disadvantage of Manchester encoding is that by replacing every data bit with two, the maximum achievable baud rate is only 50% of what the RF link would be capable of carrying. However, this is a small price to pay for the resulting data integrity as well as the datastream averaging to zero. Manchester encoding, which is a part of the IEEE 802.3 standard, is widely used from keyless entry fobs to Ethernet.

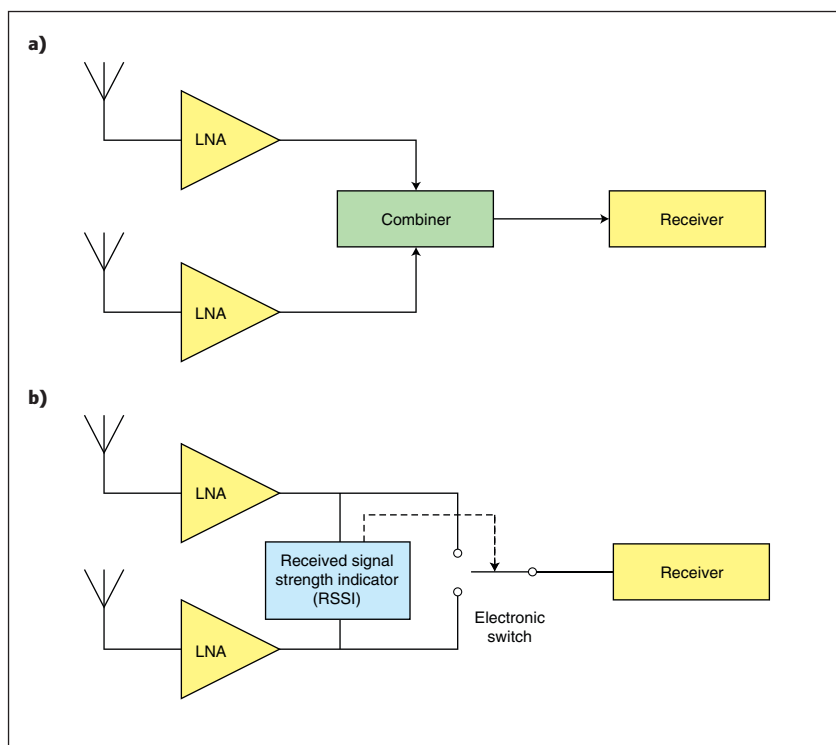
Manchester encoders and decoders used to be available as dedicated ICs, but today the encoding and decoding is mostly performed by software or FPGAs. Using software to encode or decode data can be quite demanding on the microcontroller's processing resources. On a couple of projects when I was using an Arduino, I had to free up its resources by relegating the Manchester encoding and decoding to a dedicated Atmel ATtiny85 microcontroller.

ENCODERS AND DECODERS

For experimentation, you can use SparkFun Electronics's WRL-10534 transmitter and WRL-10532 receiver, which I mentioned in previous parts of this article series. You can also use a Holtek Semiconductor HT12D decoder and HT12E encoder or Freescale Semiconductor MC145026 and MC145027 encoder and decoder pairs. These Manchester encoders and decoders are used for keyless entry with a small data-handling capability.

If you only need to transfer four bits of data at a time at a rather slow repetition rate, these chips will do just fine. Holtek Semiconductor and Freescale Semiconductor decoders and encoders enable you to transfer four bits of data. That is 16 values for the HT12D decoder and the HT12E encoder. The MC145026 encoder combined with the MC145027 decoder also provides four bits of data, but on a trinary base (high, low, floating), representing 81 combinations.

For example, if security or accidental activation by a strange transmitter is not your concern, you can use just two address bits for

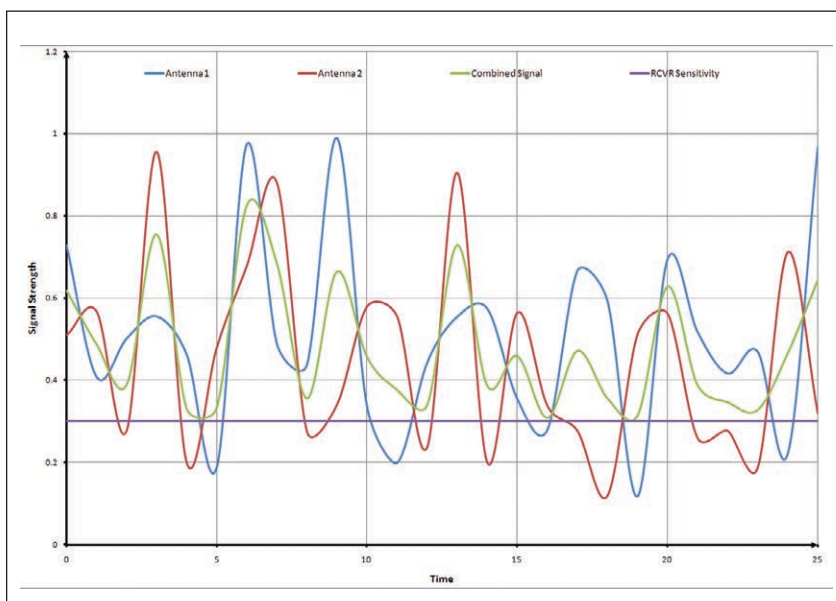


identification. You will be left with five trinary data bits using Freescale Semiconductor decoders and encoders, representing 2,187 combinations, or 10 binary bits using Holtek Semiconductor decoders and encoders, which represents 1,024 combinations. Unfortunately, the decoder ICs are built to require five or eight address bits, respectively. You will have to use a microcontroller to decode the data, which is a fairly straightforward task.

If you need full-blown data communications and are not striving for the lowest cost, don't waste your time designing your own encoder

FIGURE 2
Additive (a) and switched (b) are two approaches to antenna diversity.

FIGURE 3
Additive antenna diversity can help overcome signal fading.



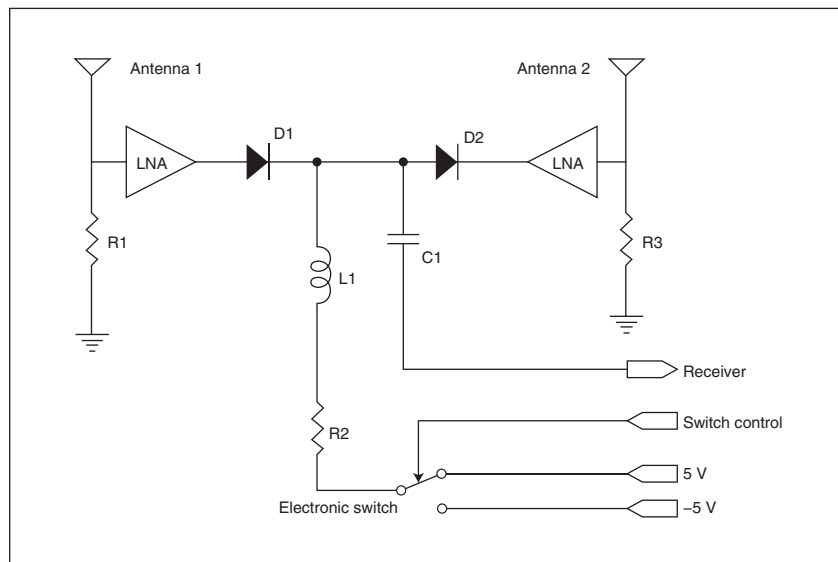


FIGURE 4

This is a typical approach to antenna switching.

and/or decoder. Instead, purchase off-the-shelf, capable, reasonably priced, reliable transmitters, receivers, or transceivers (e.g., Digi International's XBee, Linx Technologies, Micrel, RF Monolithics, etc.). Adafruit Industries is also a good source. They work right out of the box and, in my experience, you save yourself a lot of frustration.

MULTIPATH DISTORTION

Multipath distortion is a scourge of wireless data communications that is very pronounced in very high frequency (VHF) and ultra-high frequency (UHF) bands, where the low-power data transmitters operate. It is caused by the receiver antenna picking up the direct wave and its reflections off objects in its path. The reflected waves reach the antenna with a phase lag due to the longer path they traveled. All the waves combine, creating peaks and valleys of the signal strength and possibly causing spots where the signal cannot be received.

Multipath distortion is very unpredictable in real life, especially when the reflecting objects (e.g., people, transmitters, or receivers) move. To avoid the effects of reflections in tests, manufacturers test their systems' range in "open field" (i.e., well above ground and in settings where there are no substantial reflecting surfaces in the path).

Several methods can minimize the effects of multipath distortion. By changing the wavelength (i.e., the carrier's frequency), the peaks and valleys of the signal strength in the space will move and the signal's integrity will improve. This can be achieved by frequency hopping or spread-spectrum modulation. (The latter is used by XBee.) Both methods will make your design more difficult and neither is 100% reliable. (Refer to R. Lacoste's article "Don't Fade Away: A Multipath Fading Experiment," *Circuit Cellar* 247, 2011.)

ANTENNA DIVERSITY

Antenna diversity is a successful, widely used technique for suppression effects of multipath distortion. This can be done by polarization diversity, radiation pattern diversity, or spatial diversity.

For polarization diversity, both horizontal and vertical polarizations are used. Radiation pattern diversity relies on different magnitudes and phases due to the antenna radiation pattern. Spatial diversity is achieved by using two or more antennas on the receiver, transmitter, or both spaced about a quarter wavelength apart. Antenna diversity has been used for many years. Today it can be seen with some Wi-Fi routers. I saw it utilized about 30 years ago by a wireless security system receiver operating at 315 MHz.

Figure 2 shows two common ways to deal with antenna diversity. For simplicity, just two antennas are shown, although some professional systems use many. Additive diversity is one common approach, where the signal from two or more antennas is amplified by low-noise amplifiers (LNAs), combined with respect to some algorithm and sent to the



circuitcellar.com/ccmaterials

RESOURCES

Atmel Corp., "Manchester Coding Basics," Application Note 9164, 2009.

M. Hebel, G. Bricker, and D. Harris, "Getting Started with XBee RF Modules: A Tutorial for BASIC Stamp and Propeller Microcontrollers," Parallax, Inc.

R. Lacoste, "Don't Fade Away: A Multipath Fading Experiment," *Circuit Cellar* 247, 2011.

M. Maggi, "Modular RF Link Using Manchester Code (2): Software," *Elektor*, 2013.

G. Novacek, "Wireless Data Links (Part 3): Receivers and Recovery," *Circuit Cellar* 286, 2014.

Wikipedia, "Antenna Diversity."

SOURCES

ATtiny85 Microcontroller

Atmel Corp. | www.atmel.com

XBee wireless RF module

Digi International, Inc. | www.digi.com

MC14502x Encoder and decoder pairs

Freescale Semiconductor, Inc. | www.freescale.com

HT12D Decoder and HT12E encoder

Holtek Semiconductor, Inc. | www.holtek.com

WRL-10534 Transmitter and WRL-10532 receiver

SparkFun Electronics | www.sparkfun.com

ABOUT THE AUTHOR

George Novacek 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. Contact him at gnovacek@nexicom.net with "Circuit Cellar" in the subject line.



receiver (see **Figure 2a**).


Figure 3 shows an example of using a simple addition. The blue and red traces are representative of Antenna 1 and Antenna 2 signals. The green trace shows their additive combination. The violet line represents the receiver sensitivity. Even though neither antenna could continuously provide reliable reception, their combined signals do.

Switched diversity operates differently (see **Figure 2b**). Once again, the antenna's outputs are amplified by LNAs. The received signal strength indicator (RSSI) measures each antenna output's characteristics and, based on some algorithm, decides which signal is better and connects it to the receiver.

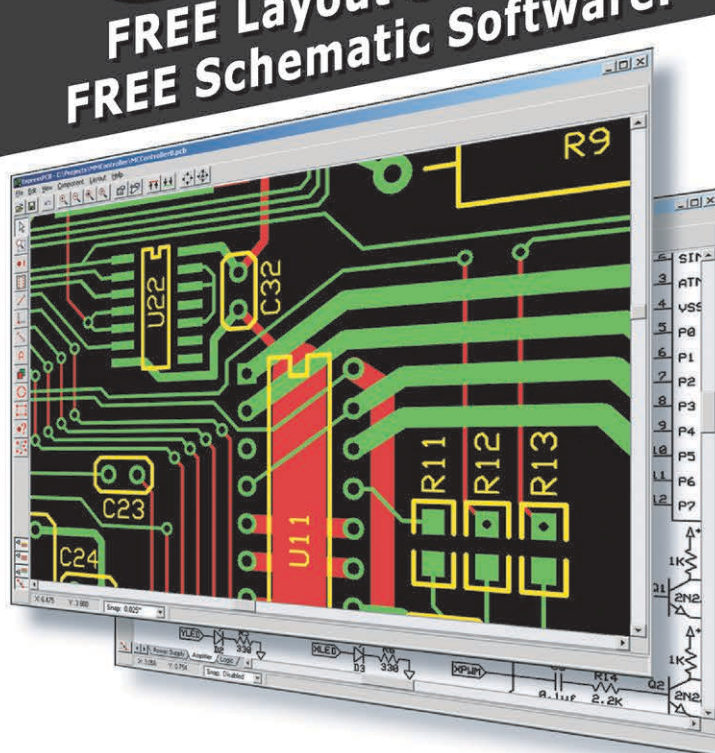
Hardware for switching between only two antennas can be quite simple (see **Figure 4**). It uses alternatively DC-biased PIN diodes to conduct or not. Inductance L1 ensures the RF signal is not loaded by the control circuitry, while C1 blocks the DC from reaching the receiver. The LNAs may not be always needed.

WIRELESS DATA EXCHANGE

This concludes this series about low-power wireless data links. It is not exhaustive, but it will give you a good start to understanding low-power wireless data exchange. In a future article I may examine important receiver issues such as intermodulation distortion.

This article's Resources section provides information for further study. If you don't feel the urge to get frustrated by building your own equipment from scratch, purchasing some of the many devices available at a reasonable off-the-shelf price will get you going quickly. 

\$51^{For 3} PCBs
FREE Layout Software!
FREE Schematic Software!

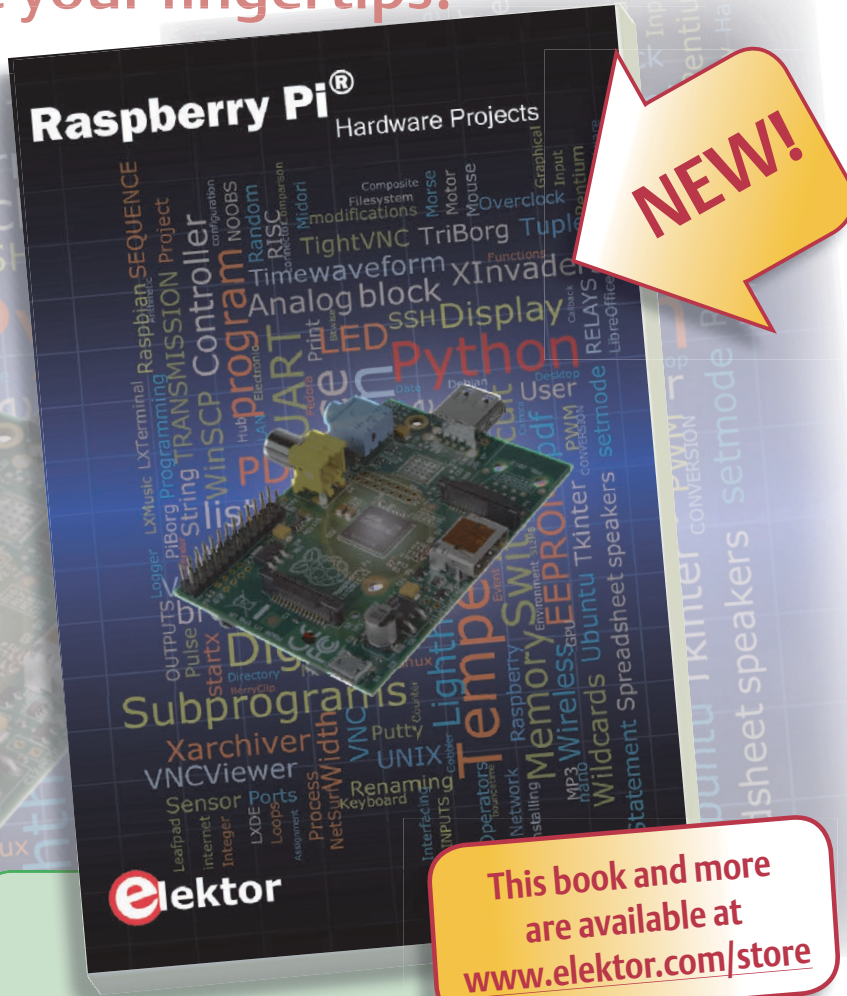


- 01 DOWNLOAD our free CAD software
- 02 DESIGN your two or four layer PC board
- 03 SEND us your design with just a click
- 04 RECEIVE top quality boards in just days

expresspcb.com

Elektor.STORE

The world of electronics
at your fingertips!



Books

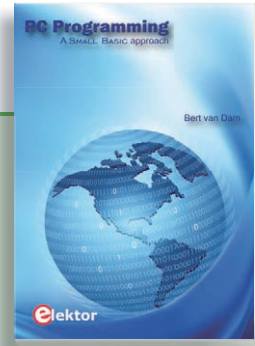
The RPi in Control Applications

Raspberry Pi Hardware Projects

This book is about the Raspberry Pi computer and its use in control applications. Dogan Ibrahim explains in simple terms, with examples, how to configure the RPi, how to install and use the Linux operating system, how to write programs using the Python programming language and how to develop hardware based projects. The book starts with an introduction to the Raspberry Pi computer and covers the topics of purchasing all the necessary equipment and installing/using the Linux operating system in command mode. Use of the user-friendly graphical desktop operating environment is explained using example applications. The RPi network interface is explained in simple steps and demonstrates how the computer can be accessed remotely from a desktop or a laptop computer. The remaining parts of the book cover the Python programming language, hardware development tools, hardware interface details, and RPi based hardware projects.

290 pages • ISBN 978-1-907920-29-5 • \$54

This book and more
are available at
www.elektor.com/store

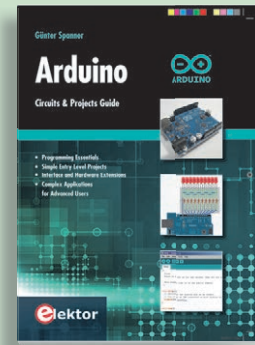


A Small Basic Approach

PC Programming

In this book we take a practical approach to programming with Small Basic. You will have an application up and running in a matter of minutes. You will understand exactly how it works and be able to write text programs, graphical user interfaces, and advanced drivers. It is so simple; you don't even need to be an adult!

194 pages • ISBN 978-1-1-907920-26-4 • \$48

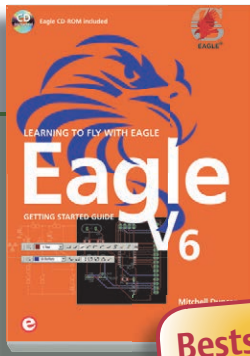


Circuits & Projects Guide

Arduino

The Arduino user is supported by an array of software libraries. In many cases, detailed descriptions are missing, and poorly described projects tend to confuse rather than elucidate. This book represents a different approach. All projects are presented in a systematic manner, guiding into various theme areas. In the coverage of must-know theory great attention is given to practical directions users can absorb, including essential programming techniques like A/D conversion, timers and interrupts—all contained in the hands-on projects. In this way readers of the book create running lights, a wakeup light, fully functional voltmeters, precision digital thermometers, clocks of many varieties, reaction speed meters, or mouse controlled robotic arms.

260 pages • ISBN 978-1-907920-25-7 • \$56

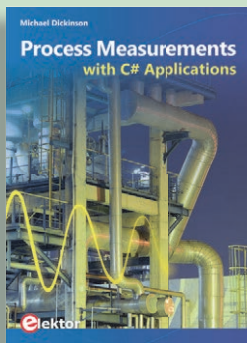


Bestseller!

Learning to Fly With Eagle Eagle V6 Getting Started Guide

The book is intended for anyone who wants an introduction to the capabilities of the CadSoft's Eagle PCB design software package. The reader may be a novice at PCB design or a professional wanting to learn about Eagle, with the intention of migrating from another CAD package. After reading this book while practicing some of the examples, and completing the projects, you should feel confident about taking on more challenging endeavors. This book is supplied with a free copy of Eagle on CD-ROM for MS Windows, Linux and Mac.

208 pages • ISBN 978-1-907920-20-2 • \$47



A Highly Practical Guide Process Measurements with C# Applications

This book introduces PC based measurement systems and software tools for those needing to understand the underlying principles or apply such techniques. C# was chosen to give readers immediate, practical experience with a prominent programming language. C-Sharp has a wide support base and is a popular choice for engineering solutions. The basics of measurement and data capture systems are presented, and examples of software post-processing

144 pages • ISBN 978-1-907920-24-0 • \$39



110 Elektor Editions, Over 2500 Articles DVD Elektor 2000 through 2009

This DVD-ROM contains all circuits and projects published in Elektor magazine's year volumes 2000 through 2009. The 2500+ articles are ordered chronologically by release date (month/year), and arranged in alphabetical order. A global index allows you to search specific content across the whole DVD. Every article is printable using a simple print function. This DVD is packed with ideas, circuits and projects that are ideal for any electronics enthusiast, student or professional, regardless of whether they are at home or elsewhere.

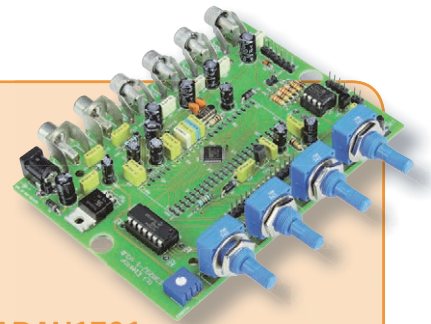
ISBN 978-1-907920-28-8 • \$121

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/store

elektor

Elektor US
111 Founders Plaza, Suite 300
East Hartford, CT 06108
USA
Phone: 860.289.0800
Fax: 860.461.0450
E-mail: order@elektor.com

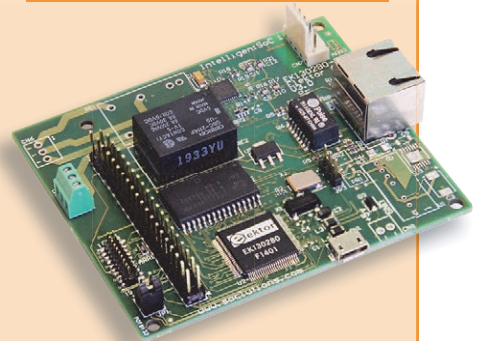


ADAU1701 Universal Audio DSP Board

Always wanted to start out with DSPs, but afraid of the SMD? Elektor is making Digital Signal Processing more fun and accessible with the release of a Universal Audio DSP Board. Based on the Analog Devices ADAU1701 DSP chip and drag-and-drop software, this board will ease your way to becoming a sound craftsman, guaranteed maths-free.

Semi-kit: all TH components, PCB with DSP preassembled

Art.# 130232-71 • \$102



E-Lock

The E-Lock chip allows you to connect your control system to the 'Network of Networks' and monitor and control it from anywhere on or around the globe using your computer, tablet or smartphone. This completely assembled module is able to establish a secure connection over the Internet (TCP/IP) using the Transport Layer Security (TLS) encryption protocol. The chip also offers a 7-line GPIO (general purpose input/output), four 16-bit ADC channels, one 12-bit DAC and one I2C bus that allow control and communication with other peripheral devices.

Ready build module

Art.#130280-91 • \$150

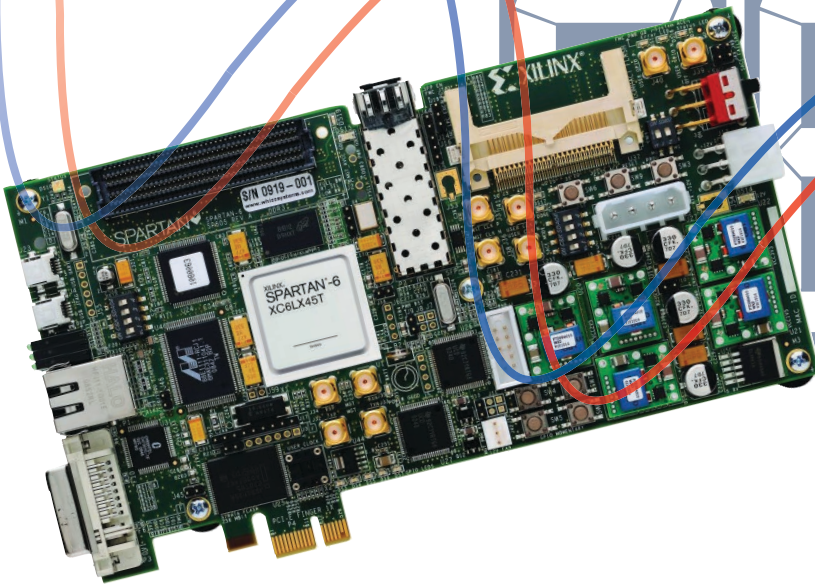
PROGRAMMABLE LOGIC IN PRACTICE

Partial FPGA Configuration

Many FPGA design attributes, such as certain clock and I/O drive settings, are not adjustable at run time. Yet in many applications, it would be convenient to adjust them during operation. This article explains how you can use partial reconfiguration to bypass these restrictions and modify FPGAs.

By Colin O'Flynn (Canada)

COLUMNS



Partial reconfiguration (PR) of an FPGA is a topic I'm sure many designers have heard about, but few have fully used. PR enables you to change part of your FPGA design during operation. It's an extremely powerful tool that can be used for very advanced topics (e.g., reloading an entire "module" in an FPGA design).

This article will only cover a more basic use, something I suspect many FPGA designers have run into. The problem is certain FPGA modules have parameters that can only be adjusted during implementation, and not at run time. You can use PR to adjust those parameters, sidestepping the issue. There are several caveats to using PR, so there is a lot to discuss.

Before I begin, I want to point you to two excellent resources. C. Cameron's article "Digital Duct Tape with FPGA Editor" (*Xilinx's Xcell Journal*, Issue 66, 2008) covers the use of the FPGA editor (which I'll briefly discuss). J. McCaskill and D. Lautzenheiser's article "FPGA Partial Reconfiguration Goes Mainstream" (*Xcell Journal* Issue 73, 2010) examines the use of PR, although it targets more advanced uses of PR.

PERFECT TIMING

I will begin by describing the exact problem I'm handling with PR. You can use PR to solve many similar problems, but having something concrete in your mind will help you understand the problem and the solution.

The digital clock module (DCM) in a Xilinx Spartan-6 FPGA has a variety of features, including the ability to add an adjustable phase shift onto an input clock.

There are two types of phase shifts: fixed and variable. A fixed shift can vary from approximately -360° to 360° in 1.4° steps. A variable shift enables shifting over a smaller range, which is approximately ± 5 nS in 30-ps steps.

Note the actual range and step size varies considerably for different operating conditions. Hence the problem: the provided variable phase shift interface is only useful for small phase shifts; any major phase shift must be fixed at design time.

To demonstrate how PR can be used to fix the problem, I'll generate a design that implements a DCM block and use PR to dynamically reconfigure the DCM.

STREAMING BITS

I used Xilinx's ISE design suite to generate a design (see **Figure 1**). I did the usual step of creating the entire FPGA bitstream, which could then be programmed into the FPGA. The FPGA bitstream is essentially a completely binary blob tells you nothing about your design. The "FPGA native circuit description (NCD)" file is one step above the FPGA bitstream. The NCD file contains a complete description of the design mapped to the blocks available in your specific chip with all the routing of signals between those blocks and useful net names.

The NCD file contains enough information for you to do some edits to the FPGA design. Then you can use the NCD file to generate a new binary blob (bitstream) for programming. A critical part of PR is understanding that when you download this new blob, you can only download the difference between the original file and the new file. This is known as "difference-based PR," and it is the only type of PR I'll be discussing.

So what's in the bitstream? The bitstream actually contains several different commands sent to the FPGA device, which instructs the FPGA to load configuration information, tells it the address information where the data is going to be loaded, and finally sends the actual data to load. Given a bitstream file, you can actually parse this information out. I've posted a Python script on ProgrammableLogicInPractice.com that does this for the Spartan-6 device. **Listing 1** shows the contents of a file that only reloads part of an FPGA.

A frame is the smallest portion of an FPGA that can be configured. The frame's size varies per device. (For the Spartan-6 I used in this article, it is 65 × 16-bit words, or 1,040 bits per frame.) You must reload the entire frame if anything inside it changes, which brings me to the first "gotcha." When using PR, everything inside that "frame" will be reloaded (i.e., parts of your design that haven't changed may become temporarily invalid because they share a configuration frame with the part of your design that has changed).

FRAMING JOB

If a frame is a collection of bits, what do those bits map to in the FPGA? At this point, things will start to get a little murky. This is partially because FPGA vendors won't tell you all the details of mapping from an NCD file to a bitstream. FPGA vendors consider the information about exact bitstream mapping to be proprietary information, which they won't publish as it may give competitors or attackers too much useful information. If you knew the mapping, it would make it easier to

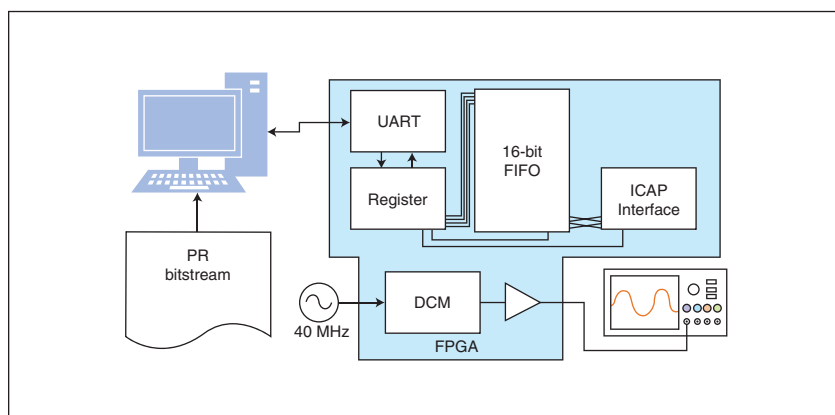


FIGURE 1

The system's general block diagram is shown. A digital clock module (DCM) block synthesizes a new clock from the system oscillator and then outputs the clock to an I/O pin. The internal configuration access port (ICAP) interface is used to load configuration data. The serial interface connects the ICAP interface to a computer via a first in, first out (FIFO) buffer.

```
FFFF: Dummy (x 8)
aa99: SYNC1
5566: SYNC2
30a1: TYPE1:WRITE to CMD 1 words
0007: DATA
2000: Nop (x 1)
31a1: TYPE1:WRITE to FLR 1 words
0380: DATA
31c2: TYPE1:WRITE to IDCODE 2 words
0400: DATA
1093: DATA
3141: TYPE1:WRITE to COR1 1 words
3d00: DATA
3161: TYPE1:WRITE to COR2 1 words
09ee: DATA
3022: TYPE1:WRITE to FAR_MAJ 2 words
0009: FAR_MAJ Data: Block 0, Row 0, Major 9
001e: FAR_MIN Data: BRAM 0, Minor 30
30a1: TYPE1:WRITE to CMD 1 words
0001: DATA
5060: TYPE2:WRITE to FDRI
0000
0082: 130 frames
...Data Frames Not Printed...
0029:
7f57:
30a1: TYPE1:WRITE to CMD 1 words
0003: DATA
2000: Nop (x 4)
3002: TYPE1:WRITE to CRC 2 words
0017: DATA
6e2c: DATA
30a1: TYPE1:WRITE to CMD 1 words
000d: DATA
2000: Nop (x 16)
```

LISTING 1

I used a Python script to dissect this partial bitstream. Several Write commands write to configuration registers, then there is a 130 data word block, which is followed by some additional commands and a cyclic redundancy check (CRC).

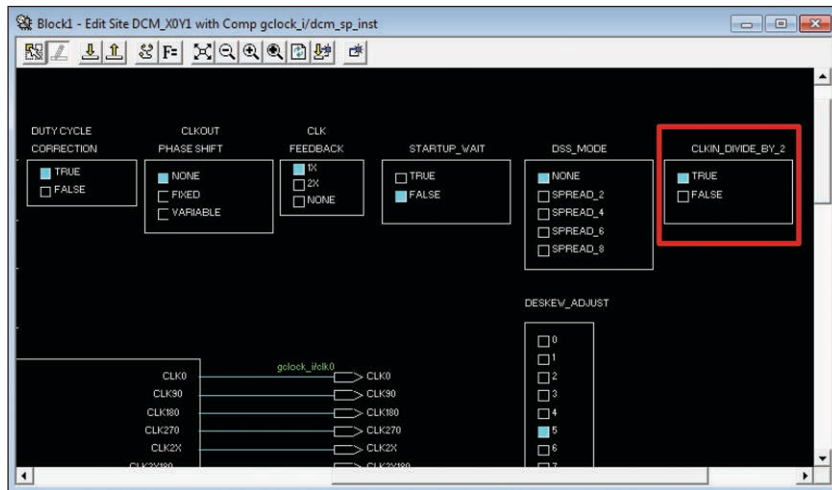


PHOTO 1

You can use the FPGA editor to modify the digital clock module (DCM) block. Here the `CLKIN_DIVIDE_BY_2` attribute is being switched to “True.” You can also display additional attributes, which enables you to change the phase shift and other settings.

reverse engineer an FPGA bitstream back into a useful higher-level system. So don't expect the mapping to become public information anytime soon.

There are several types of frames. For Spartan-6 three types are valid: Type 0 for core features (CLB, DSP, I/O interconnect, and clocking), Type 1 for block RAM, and Type 2 for I/O blocks. I'll mainly be examining Type 0 frames. The number of frames also varies by device, as larger devices require more frames. For the Avnet Spartan-6 LX9 MicroBoards I used, the device requires 2,028 Type 0 frames.

Note that each frame contains information about multiple Spartan-6 configurable logic blocks (CLBs) and configuring a single CLB requires multiple configuration frames. Certain small aspects of a CLB, such as a

single look-up-table (LUT), or special features such as the DCM will only require a single frame for adjustment. However, this single frame will still be reconfiguring other parts of the FPGA. For example, you cannot “mask off” to reconfigure only a single LUT.

MODIFYING A DESIGN

To begin with, I implemented the design as usual. Then I loaded the NCD file in the FPGA editor. The FPGA editor (accessible in Xilinx ISE from Tools > FPGA Editor > Post-Place & Route) enables you to perform many design modifications. I'll relegate step-by-step instructions for this modification to my companion post on ProgrammableLogicInPractice.com. You can also refer to the previously mentioned “Digital Duct Tape with FPGA Editor” article.

In this example I modified the `CLKIN_DIVIDE_BY_2` parameter of the DCM block used in my design (see **Photo 1**). Then I saved a new NCD file, which was simply the original file but with the one parameter modified. Finally, I used the `bitgen` command-line utility to generate a “difference” file. **Listing 2** shows the command sequence.

A second “gotcha” for the Spartan-6 device is that you may receive an error about 9-Kb block RAM initialization data being incompatible with partial bitstreams. This requires you to add the `-convert_bram8` switch in the Xilinx ISE as one of the special command-line MAP arguments. The details are available at ProgrammableLogicInPractice.com.

Again I must stress that this difference file contains a complete copy of the configuration frame(s) where changes occurred. You need to regenerate the difference file every time you change the design. There may be other logic or features in that same configuration frame, not just the feature you are trying to modify.

To simplify this process, you can save a script that applies the necessary changes to the NCD file and saves the results. While manually changing the NCD file in the FPGA editor, you simply record your actions to a script. As described in Cameron's “Digital Duct Tape with FPGA Editor,” you can clean up this script file a bit and then run it as a command-line argument to the FPGA editor. The result is that whenever you build a new design, it's trivial to generate the required partial bitstream(s).

You can even use another script in Russian nesting doll fashion to create the FPGA editor script. For example, this master script can generate scripts with combinations of various options. I used this to produce partial bitstreams for every phase shift from -255 to 255, giving me 511 differential bitstreams. I could use any of them to load a specific phase

```
#Generate difference from original and edited (i.e.
apply changes)
bitgen -g ActiveReconfig:Yes -r original.bit -w new.ncd
diffbits.bit -b
```

```
Creating bit map...
Loading bitfile original.bit...
Saving bit stream in “diffbits.bit”.
Making partial bitfile...
There were 1 frames different between diffbits.bit and
new.ncd; UserID=0xFFFFFFFF.
Saving bit stream in “diffbits.rbt”.
Bitstream generation is complete.
```

```
#Generate difference from edited and original (i.e. undo
changes)
bitgen -g ActiveReconfig:Yes new.ncd new.bit
bitgen -g ActiveReconfig:Yes -r new.bit -w original.ncd
diffbits.bit -b
```

LISTING 2

The first `bitgen` command also shows an example output. Note the number of frames in difference is explicitly printed. If you are modifying multiple items, you can experimentally determine whether or not they are saved in the same configuration frame.

shift on the DCM block.

To avoid any unexpected surprises, you will likely wish to lock down the location of the DCM block (or other resource you'll be modifying). This can be done through the standard constraints (.UCF) file.

EXTERNAL SURGERY

Initially, you may wish to load partial configuration files through the JTAG connection. If you are using a development board that provides a true USB-JTAG interface, this should be trivial, as the Impact tool supports partial bitstreams. I say "should" because, in my experience it often does not work with the Spartan-6 device. This is something that multiple (unanswered) questions by other users confirm. This boils down to PR on the Spartan-6 devices being officially unsupported.

DIY SURGERY

Xilinx devices provide the internal configuration access port (ICAP) for internally loading the device bitstream. The ICAP interface is an opaque block you can interface from your code (see **Figure 2**). Note that the ICAP interface and external interfaces are internally MUXed together into a single configuration interface. Therefore, attempting to access the FPGA using the external connection while

the ICAP interface is loading a new bitstream may cause problems. This is explained in more detail in the application notes specific to each family. For the Spartan-6, this is Xilinx's "Spartan-6 FPGA Configuration User Guide" (see Resources).

So how can you load data into the ICAP device? As an example, I'm providing a very dumb interface. A simple first in, first out (FIFO) buffer is loaded with commands from the computer. When a Go command is received, the system loads everything into the ICAP port. It assumes your connection from the computer is not subject to corruption (i.e., another layer provides protection), although if you wanted to improve the module I'd suggest calculating a cyclic redundancy check (CRC) of the data buffer and verifying it before reloading everything.

While this may seem obvious, I have to



ABOUT THE AUTHOR

Colin O'Flynn (coflynn@newae.com) has been building and breaking electronic devices for many years. He is currently completing a PhD at Dalhousie University in Halifax, NS, Canada. His most recent work focuses on embedded security, but he still enjoys everything from FPGA development to hand-soldering prototype circuits. Some of his work is posted on his website at www.colinoflynn.com.



Sign up today and **SAVE 50%** • Sign up today and **SAVE 50%** • Sign up today and **SAVE 50%**

Now offering student **SUBSCRIPTIONS!**

When textbooks just aren't enough, supplement your study supplies with a subscription to *Circuit Cellar*. From programming to soldering, robotics to Internet and connectivity, *Circuit Cellar* delivers the critical analysis you require to thrive and excel in your electronics engineering courses.

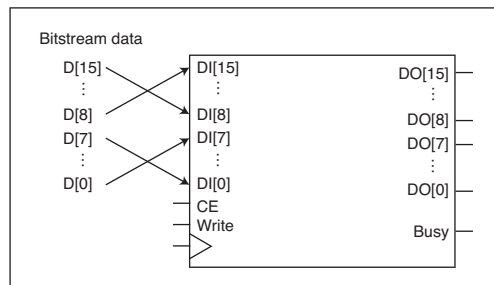
Sign up today and **SAVE 50%** • Sign up today and **SAVE 50%** • Sign up today and **SAVE 50%**

www.circuitcellar.com/subscription



FIGURE 2

The internal configuration access port (ICAP) block simply requires you to feed in the bitstream data. Pay attention to some special bit swapping required, as detailed in Xilinx's "Spartan-6 FPGA Configuration User Guide."



stress that you don't reload part of the design that deals with the ICAP interface! This is more difficult than you may think when using Spartan-6 devices. You cannot tell the tools to avoid routing any connections in certain blocks. This is primarily because PR isn't officially supported for the Spartan-6 devices. Thus when generating the PR bitstreams, it's important to verify what exactly is being modified.

To load the files into the ICAP interface, you can use the `bitgen` utility to generate a "raw bits" (.rbt) file. The 16-bit per line "raw bit" file is in ASCII text. It contains the same data as the bitstream (.bit) file, but is slightly easier to hack together a quick program or script to read the file.

I won't go into all the details in this article. This example project sends the .rbt file's contents to a FIFO on the FPGA. The FPGA's FIFO is simply 16 bits wide. The .rbt file is copied line-for-line into the FIFO. Once the file is downloaded, the data is dumped into the ICAP interface. Be aware there are some differences in bit representations in the .rbt file and what the ICAP is expecting, as detailed in Xilinx's Spartan-6 FPGA Configuration User Guide or similar for your FPGA family. **Figure 1** shows a system block diagram.

COMPLETE EXAMPLE

I can't describe a complete project in detail in this article. Instead I'm going to redirect you

to ProgrammableLogicInPractice.com, where I've used a Spartan-6 LX9 device to complete a sample PR. I provided two examples of using PR. First, a DCM is dynamically reconfigured to alter certain parameters that are normally fixed at design time. Second, a clock output's drive current is switched from the 12-mA default to either 2 or 24 mA. **Figure 3** shows the results of loading these partial bitstreams into the running system.

This raises an interesting point. It's possible to load multiple partial bitstreams to combine effects. For example, I can generate three bitstreams: an output at 2 mA, an output at 24 mA, and a clock at half the regular rate. Provided each of those partial bitstreams modifies a separate configuration frame, it's possible to load the half-speed clock partial file and then the 24-mA drive partial file. I never needed to generate a partial bitstream that contained both of the modifications together. However, if one configuration frame covers two options I wish to separately modify, I need to generate partial bitstreams for every combination of the two options.

While discussing partial bitstreams, remember you may need to generate a partial bitstream to undo your changes (see **Listing 2**). Doing so requires that you first generate a full bitstream *with* the changes in it, and then generate a partial bitstream with the changed version as the "reference" bitstream and the original .NCD file as the new bitstream.

This brings me to another "gotcha." If you're running a script to generate many modified bitstreams (e.g., phase setting of -255, -254, -253 ... 254, 255), you need to be very careful with the "reference" you're generating the difference against.

In particular, say your default design has a zero phase shift. This means when your script generates a difference between the file when the phase is set to zero and the original file, it will show that no frames are different. If you don't catch this and later attempt to use the partial bitstream file, you'll discover the file doesn't reset the setting back to zero. Instead it leaves the setting as is. This makes sense to the tools, since they are assuming the current design had a zero phase shift, so there was no need to overwrite the configuration with zero again. Of course, if you were hoping to use this bitstream to return from any phase shift to zero, it won't work.

RECONFIGURATION COMPLETE

I hope I've shed a tiny bit of light onto using a difference-based PR on Spartan-6 devices. If you are targeting devices that Xilinx officially supports with PR, you have a lot of



circuitcellar.com/ccmaterials

RESOURCES

C. Cameron, "Digital Duct Tape with FPGA Editor," *Xcell Journal*, Xilinx, Inc., Issue 66, 2008.

J. McCaskill and D. Lautzenheiser, "FPGA Partial Reconfiguration Goes Mainstream," *Xcell*

Journal, Xilinx, Inc., Issue 73, 2010.

ProgrammableLogicInPractice.com

Xilinx, Inc., "Spartan-6 FPGA Configuration User Guide," UG380, 2013.

SOURCES

LX9 MicroBoard

Avnet, Inc. | www.avnet.com

Spartan-6 FPGA family and ISE design suite

Xilinx, Inc. | www.xilinx.com

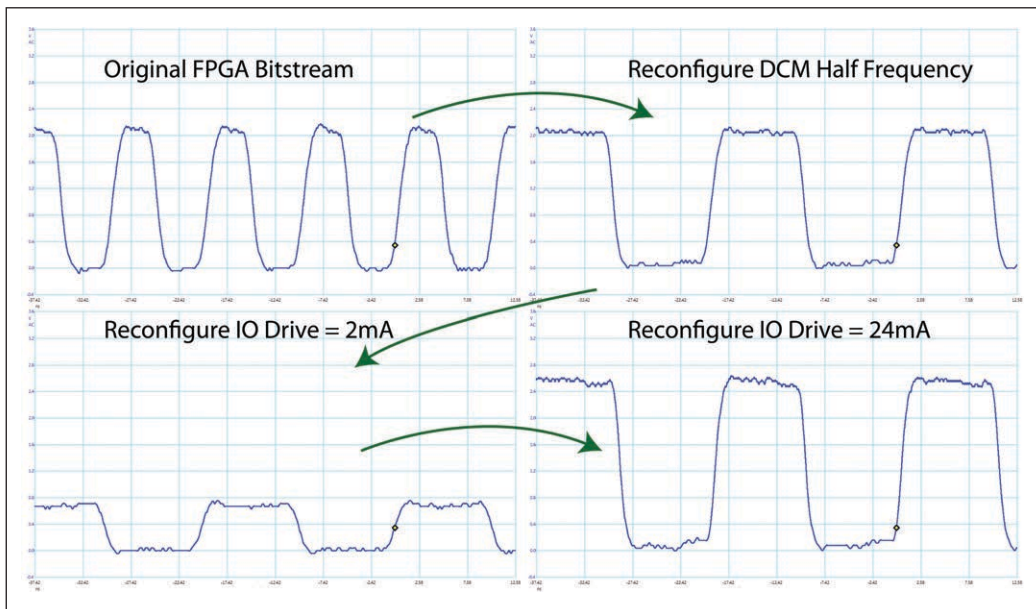


FIGURE 3

The output of a digital clock module (DCM) block is routed to an FPGA pin. Using partial reconfiguration (PR), the FPGA design can be tweaked without having to reconfigure the complete design. A DCM attribute is used to divide the input frequency by half in the first reconfiguration. The I/O pin's "drive" attribute is changed to 2 mA, and then finally to 24 mA.

additional information and some nicer tools at your disposal. However, for many applications the simple approach I demonstrated in this article will still be useful. For example, switching I/O standards or drive strengths on I/O lines could help your product handle cases where a single connector is expected to support a range of devices

I posted more information on ProgrammableLogicInPractice.com, which includes a video of the this project running. You can also see an example of how I integrated PR into my open-source ChipWhisperer project, which uses PR to dynamically program a phase shift into the DCMs inside the FPGA. [E](#)

COLUMNS

engineering embedded
 DISCUSS social media
 information design tips tutorial software
 engineering tools
 audio business talk
 electronics media
 system data COMMUNITY
 NETWORKING
 product news engage
 projects

Want to talk to us directly?
 Share your interests and opinions!
 Check out our New Social Media Outlets
 for direct engagement!

CIRCUIT CELLAR / AUDIOXPRESS / ELEKTOR

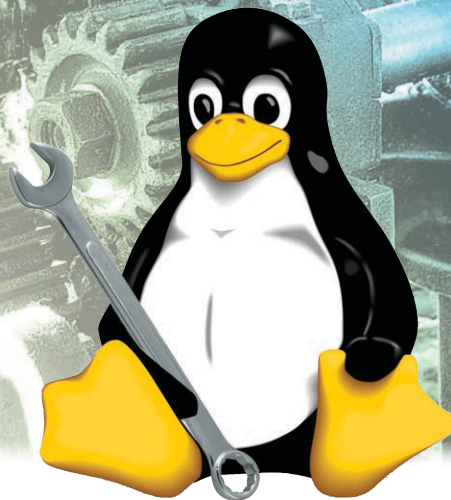
EMBEDDED IN THIN SLICES

Linux System Configuration (Part 1)

The Linux Kernel

Configuring Linux for embedded systems doesn't have to be complicated. While this series won't include design details, it will provide information about the available possibilities. This article focuses on Linux kernel configuration.

By Bob Japenga (US)



One of the embedded systems my company designed is being deployed in ways in which it was not intended. I am sure that never happens to you. In fact, we specifically told customers not to install these systems in this particular way. Since you can never tell a customer he is wrong for not reading your instructions, nor can you ask him to reinstall thousands of units, we were looking into software changes that would help us survive this installation problem.

We determined that if we could reduce the overall power the unit draws, we might be able to mitigate the problem. Our customer's chief technical officer asked if we could put the processor in Sleep mode to reduce power draw during idle times. I knew this could be done with the hardware, but I did not know if our Linux kernel was configured to do so.

Linux kernels have hundreds of parameters you can configure for your specific application. Since Sleep mode is used mostly on battery-powered units, we had not configured the kernel to support it. Although we can update the kernel remotely, it has a costly cellular data plan. (We have a 1-MB per month data plan. The kernel update would be approximately 3 MB.) So because we didn't adequately plan ahead, this solution was not viable.

Hopefully this article will help you plan ahead. Many of the options I discuss cost little in terms of memory and real-time usage. This article will examine the kinds of

features that can be configured to help you think about these things during your system design. At a minimum, it is important for you to know what features you have configured if you are using an off-the-shelf Linux kernel or a Linux kernel from a reference design. Of course, as always, I'll examine this only in thin slices.

WHY CONFIGURE THE KERNEL?

Certainly if you are designing a board from scratch you will need to know how to configure and build the Linux kernel. However, most of us don't build a system from scratch. If we are building our own board, we still use some sort of reference design provided by the microprocessor manufacturer. My company thinks these are awesome. The reference designs usually come with a prebuilt kernel and file system.

Even if you use a reference design, you almost always change something. You use different memory chips, physical layers (PHY), or real-time clocks (RTCs). In those cases, you need to configure the kernel to add support for these hardware devices. If you are fortunate enough to use the same hardware, the reference design's kernel may have unnecessary features and you are trying to reduce the memory footprint (which is needed not just because of your on-board memory but also because of the over-the-air costs of updating, as I mentioned in the introduction).

Or, the reference design's kernel may not have all of the software features you want.

For example, imagine you are using an off-the-shelf Linux board (e.g., a Raspberry Pi or BeagleBoard.org's BeagleBone). It comes with everything you need, right? Not necessarily. As with the reference design, it may use too many resources and you want to trim it, or it may not have some features you want. So, whether you are using a reference design or an off-the-shelf single-board computer (SBC), you need to be able to configure the kernel.

LINUX KERNEL CONFIGURATION

Many things about the Linux kernel can be tweaked in real time. (This is the subject of a future article series.) However, some options (e.g., handling Sleep mode and support for new hardware) require a separate compilation and kernel build. The Linux kernel is written in the C programming language, which supports code that can be conditionally compiled into the software through what is called a preprocessor `#define`.

A `#define` is associated with each configurable feature. Configuring the kernel involves selecting the features you want with the associated `#define`, recompiling, and rebuilding the kernel.

Okay, I said I wasn't going to tell you how to configure the Linux kernel, but here is a thin slice: One file contains all the `#defines`. Certainly, one could edit that file. But the classic way is to invoke `menuconfig`. Generally you would use the `make ARCH=arm menuconfig` command to identify the specific architecture.

There are other ways to configure the kernel—such as `xconfig` (QT based), `gconfig` (GTK+ based), and `nconfig` (ncurses based)—that are graphical and purport to be a little more user-friendly. We have not found anything unfriendly with using the classical method. In fact, since it is terminal-based, it works well when we remotely log in to the device.

Photo 1 shows the opening screen for one of our configurations. The options are reasonably well grouped to enable you to navigate the menus. Most importantly, the mutual dependencies of the `#defines` are built into the tool. Thus if you choose a feature that requires another to be enabled, that feature will also automatically be selected.

In addition to the out-of-the-box version, you can easily tailor all the configuration tools if you are adding your own drivers or drivers you obtain from a chip supplier. This means you can create your own unique

menus and help system. It is so simple that I will leave it to you to find out how to do this. The structure is defined as `Kconfig`, for kernel configuration (see Resources for more information).

HARDWARE OPTIONS

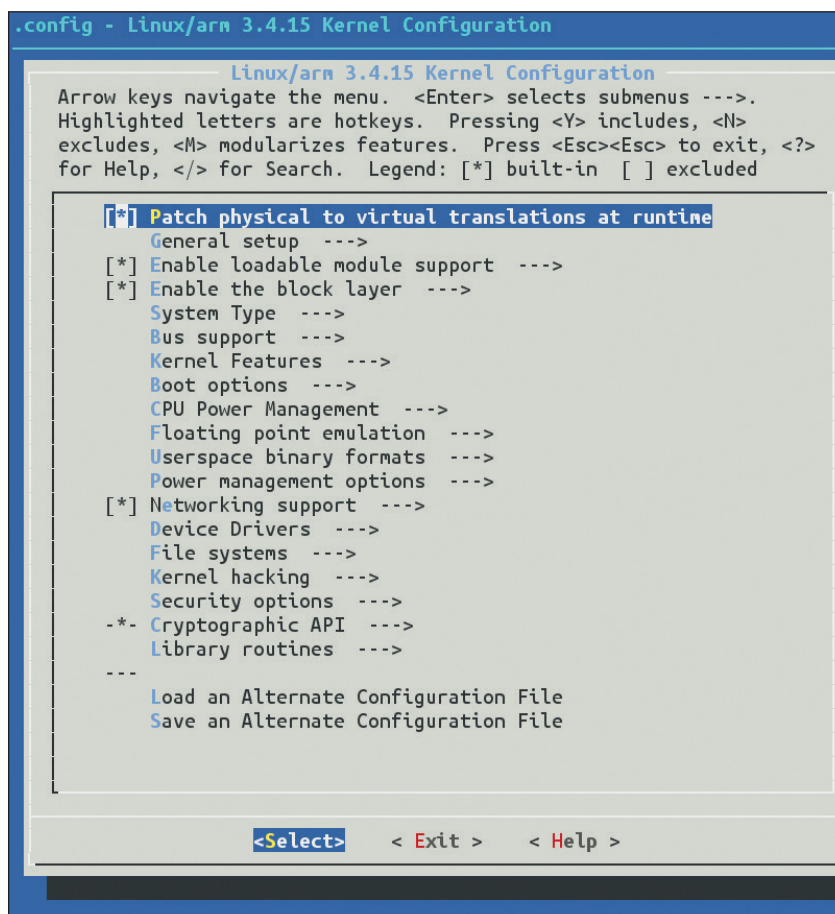
As I have already mentioned, one of the chief reasons for configuring the kernel is to select the specific hardware for which you are building the kernel. It starts with the processor and/or system type. For example, ARM's ARM9 family of processors, which my company heavily uses, has a dizzying array of options.

`menuconfig` walks you through those options. You can always learn more about an option by selecting the context-sensitive help available for each option. Specific features (e.g., your chipset's power management) can be configured.

For example, the different level of caching on the ARM processor—I-cache (instruction) and D-cache (data)—can be configured. Also you can choose the kind of frequency throttling you want to take place to conserve power. (Check out how often the clock frequency is changed on your smartphone sometime. The algorithms to conserve power are amazing.)

PHOTO 1

This opening screen includes well-grouped options for easy menu navigation.



Once you have specified the processor and system type, you need to select the specific hardware drivers you want loaded. Before you do that, you should decide if you want the ability to maintain “loadable module support.” If you don’t choose this option, all drivers will be part of the kernel image.

Although it may seem better to keep things fixed in your embedded system, enabling modules to be loadable lets you update a driver without updating the entire kernel. We like to make every driver possible loadable to enable us to easily replace one small file rather than requiring a change to the kernel image. Loadable driver modules are stored on the file system, whereas embedded drivers are stored as part of the kernel image. Loadable modules can help you considerably enhance your system’s features without expanding the kernel partition.

`menuconfig` enables you to select from an array of memory devices, RTCs, Ethernet PHYs, keyboards, mice, joysticks, watchdog timers, I/O expanders, ADCs, DACs, sound modules, I²C and SPI EEPROMs, 802.11 wireless options, Bluetooth options, touchscreen and display drivers, and many more. Once selected, some drivers can configure certain features. For example, the serial driver may support direct memory access (DMA). This can enable you to transport data at much

higher than 115.2 kbps.

When we are designing a board and need to add new hardware, we start in `menuconfig` to determine what hardware is already supported. Most of our projects don’t give us the leisure to develop a driver from scratch. Encourage your hardware designer to become familiar with the options available from `menuconfig` and you will save a lot of future grief.

THE SOFTWARE

Not only can you configure the Linux kernel with respect to the hardware, you can choose what software features you want to include. One of the most important features is the file systems you want to support.

As I discussed in previous articles, Linux supports several flash file systems. You need to select one or more you think you will be

utilizing, including the network file system (NFS).

Security is an ever-increasing concern and Linux is far and away the most powerful tool we have in our arsenal to provide the kind of security our customers demand. Working with security algorithms is like wading in alphabet soup: ECB, LRW, PCBC, XTS, HMAC, and XCBC cryptographic schemes are available.

Various algorithms are also available for cryptography including: SHA, MD5, Blowfish, RIPEMD, AES cipher, ARC cipher, DES, triple DES, and many others. Do you need all of these? Most likely not, but they don’t take a lot of resources, and you never know when you will need which part of the alphabet!

Certain system features can be configured including System V IPC (see my article “Concurrency in Embedded Systems Part 6: POSIX FIFOs and Message Queues,” *Circuit Cellar* 273, 2013), POSIX message queues, and support for triggering timers, signals, and events from file activity. Most embedded systems did not previously need to support plug-in modules. However, with the advent of USB ports, all of our embedded systems with USB ports permit hot loading of devices via USB.

Point-to-Point Protocol (PPP) is another helpful option my company uses with all our modems (both cell modems and dial-up modems as a configuration item). Support for IPv6 is also a kernel option you will want

“Although it may seem better to keep things fixed in your embedded system, enabling modules to be loadable lets you update a driver without updating the entire kernel.”



circuitcellar.com/ccmaterials

RESOURCES

3C Portal, “Applications and Tools for Android,” www.3c71.com.

ARM, Ltd., “ARM Infocenter,” <http://infocenter.arm.com>.

B. Japenga, “Concurrency in Embedded Systems Part 6: POSIX FIFOs and Message

Queues,” *Circuit Cellar* 273, 2013.

The Linux Kernel, “Documentation Extracted from the Linux Kernel and Mirrored on the Web Where Google Can Find It,” www.kernel.org/doc.

Real-Time Embedded, “Working with Kconfig,” www.rt-embedded.com.

SourceForge, “xconfig,” <http://sourceforge.net>.

Wikipedia, “List of Algorithms.”

SOURCES

ARM9 Family of microprocessors
ARM, Ltd. | www.arm.com

BeagleBone Linux computer
BeagleBoard.org | www.beagleboard.org

to provide, even if you are currently using IPv4. We are quickly running out of 32-bit IP addresses, so IPv6 will be required before we know it.

A very important feature for us in designing real-time systems is the preemption models available. Through menuconfig you can choose to use various forms of preemption. Using the full real-time control, we can write user space threads with guaranteed determinacy in the sub-millisecond range.

Last but not least, the kernel enables you to add a significant number of debugging features. In most cases these will only be used during development because of their overhead in RAM and real time. However, through configuration, you can enable kernel core dump tracing, stack dumps, and kernel-level printf support.

Serial analyzers for SPI, I²C, USB, and RS-232 are available, if enabled. We often instrument portions of the kernel during debug and then turn these features off prior to release for verification.


KNOW YOUR CONFIGURATION

Even if you are using a reference design or an off-the-shelf Raspberry Pi, as a system

ABOUT THE AUTHOR

Bob Japenga has been designing embedded systems since 1973. In 1988, along with his best friend, he started MicroTools, which specializes in creating a variety of real-time embedded systems. With a combined embedded systems experience base of more than 200 years, they love to tackle impossible problems together. Bob has been awarded 11 patents in many areas of embedded systems and motion control. You can reach him at rjapenga@microtoolsinc.com.



designer, you should be aware of what extra overhead is being configured into your kernel and know what features are missing that may be needed to develop impressive systems. Hopefully this article gave you a thin slice of how to do this. 

COLUMNS

RS485/422/232/TTL

ASC24T

RS232<=>RS485 ATE Converter

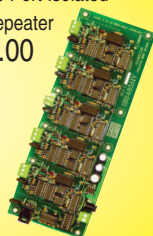


\$45.00 board only

IBS485HV

5 Port Isolated

RS485 Repeater
\$349.00



Enclosures, Cables,
Power Supplies and Other
Accessories



INTRODUCING THE SMFCOMX!

- Converters
- Repeaters
- Multi-Repeaters
- Hubs
- Fiber Optics
- Isolators
- Extended Distance Units
- Serial to Digital I/O
- Large Multi-Drop Networks
- Custom Units & Smart Units
- Industrial, 3.0 KV Isolation



Call the RS485 Wizards
513-874-4796
www.rs485.com

SCOPE DEALS

6-IN-1 SCOPE

2-ch 2MSa/s 200kHz 11-bit oscilloscope, 2-ch spectrum analyzer, 2MSa/s 8-bit arbitrary waveform generator, function gen, network analyzer, PWM/Digital I/O. - CGM-101 **\$99.95**



PASSPORT-SIZE PC SCOPES

Great scopes for field use with laptops. Up to 200MHz bandwidth with 1GSa/s, high speed data streaming to 1MSa/s, built-in 1GSa/s AWG/wfm gen. - PS2200A **\$179+**



30MHz SCOPE

Remarkable 30MHz, 2-ch 250MS/s sample rate oscilloscope. 8-in color TFT-LCD and AutoScale function. Includes FREE carry case + 3 yr warranty! - SDS5032E **\$289**



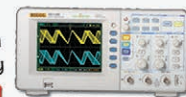
60MHz SCOPE

Best selling 60MHz 2-ch scope with 500MSa/s rate + huge 10MSa memory! 8" color TFT-LCD. Includes FREE carry case! - SDS6062 **\$349**



100MHz SCOPE

High-end 100MHz 2-ch 1GSa/s benchscope with 1MSa memory and USB port + FREE scope carry case. New super low price! - DS1102E **\$399**



INCREDIBLE LOW PRICES, FREE TECHNICAL SUPPORT
GREAT CUSTOMER SERVICE **SAELIG.COM**

THE DARKER SIDE

VOLTAGE

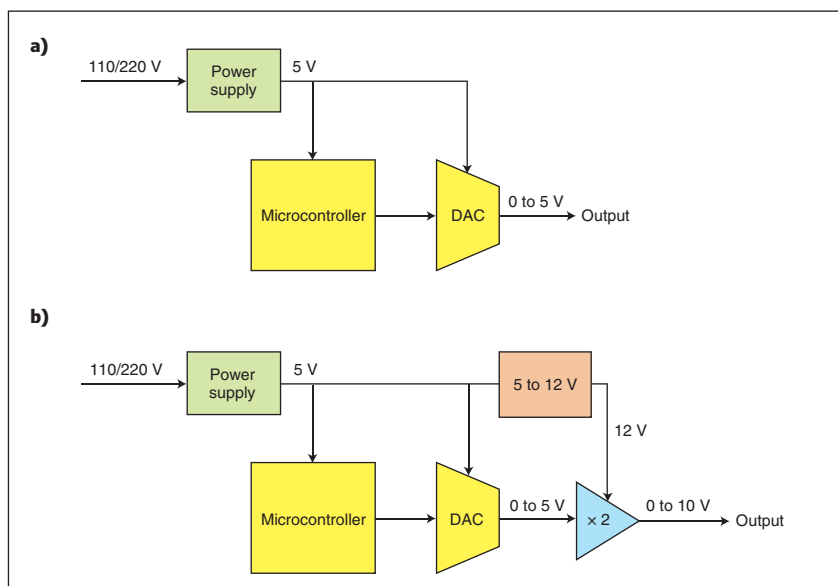
Step-Up Techniques

Most engineering projects require a power supply. Step-up circuits can help increase your voltage. This article discusses design options including charge boost converters, charge pumps, transformers, and voltage multipliers.

By Robert Lacoste (France)

FIGURE 1

a—This block diagram shows a typical 5-V powered signal-generator device. **b**—If the specification changes and requires a 0-to-10-V output, you will need to change the power supply or add a 5-to-12-V step-up circuit.



Welcome back to the Darker Side. Every designer has to find optimal solutions to engineering problems. For an electrical engineer, this could include complex high-speed processing systems, RF top-notch devices, or next-century teleportation projects. These problems all share one common subsystem: They need a power supply. Power usually comes from a battery, a brick-wall transformer, or an embedded

AC/DC converter. This power source delivers one or several voltage outputs, which are used in the design either directly or through secondary voltage regulators or DC/DC step-down converters (often called point-of-load converters).

In an ideal world, you will have designed the system well from the start, listed its power supply requirements, and then designed the power supply accordingly. However, experience shows that life is usually a bit trickier, and you may discover that you require a voltage higher than all the available power supplies.

Let me provide an example. Imagine you designed a signal-generator device. The specification stated that the output signal should range from 0 to 5 V, so you used a single low-voltage source, such as a 5-V DC brick-wall supply (see **Figure 1**). A couple of days before delivery, the customer told you that the output must be 0 to 10 V to make it compatible with an external device.

Adding an op-amp stage with a 2x gain is not a problem, but it will require a voltage source above 10 V even if you only need a couple of milliamps. If you don't want to change the external power supply, there are only two options: say no to your customer (which may not be the best long-term solution) or add an on-board voltage step-up

circuit, which would increase the available 5 V to about 12 V. What are your design options?

READY-MADE BOOST CONVERTERS

Let's start with the easiest solution. You can simply buy a step-up voltage boost module and plug it into your design. This solution may be expensive, but it would solve the problem. As an example, Traco Electronic provides single inline packages with many voltage and power versions. **Photo 1** shows the TME0512S DC/DC converter, which would be a good choice. The compact (6-mm x 10-mm x 12-mm) voltage converter has a 4.5-to-5.5-V input, a 12-V output, and 1 W available power.

This solution could be the best one for your project, but has some downsides. First, it would add \$5 to your bill. Second, you probably don't have this component on your shelf, so you will need to order it, crossing your fingers that you receive it soon enough. This could also be more difficult if you require exotic I/O voltages. Finally, this type of step-up converter is built around a DC/DC boost converter, which may generate spurious frequencies that can interfere with your sensitive analog sections.

Once again, such a brick converter may be the best solution for your project, especially if you require a significant output power. What alternatives are available if you need a small, inexpensive current?

CHARGE PUMPS

The easiest, least expensive, and cleanest way to generate a small amount of above-the-rail voltage is to implement a capacitor charge pump. The idea is simple. Imagine you charge a capacitor to the supply voltage. Then you disconnect it and reconnect its negative terminal to the positive supply voltage rail. Since the capacitor is still charged, the voltage between its positive terminal and the ground will be twice the supply voltage. If you transfer this voltage to another capacitor and keep cycling you will have a charge pump (see **Figure 2**).

Plenty of dedicated charge pump chips are available online, but what is the simplest solution if you already have a microcontroller on board? **Figure 3** shows the answer. It requires just two diodes and two capacitors. The idea is to use one microcontroller I/O pin to generate a square wave (e.g., in the kilohertz range). This could be accomplished with a microcontroller's hardware PWM or a firmware-based timer and I/O pin toggling.

When this PWM signal is low, the first capacitor C1 is charged through D1 and R1. R1 helps limit the current drawn on the

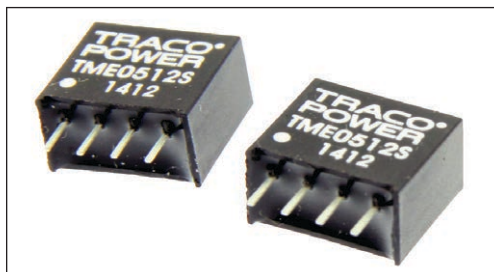


PHOTO 1

Ready-made miniature DC/DC step-up converters are available off the shelf. For example, Traco Electronic's TME0512S DC/DC converters generate 1 W and use very little board space.

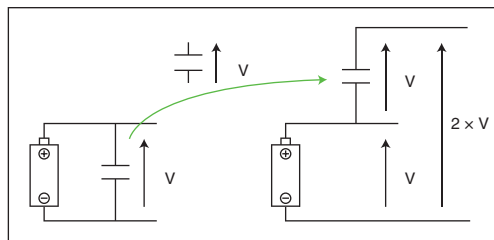


FIGURE 2

A charge pump's basic principle is shown. You charge a capacitor then add its voltage to the supply rail.

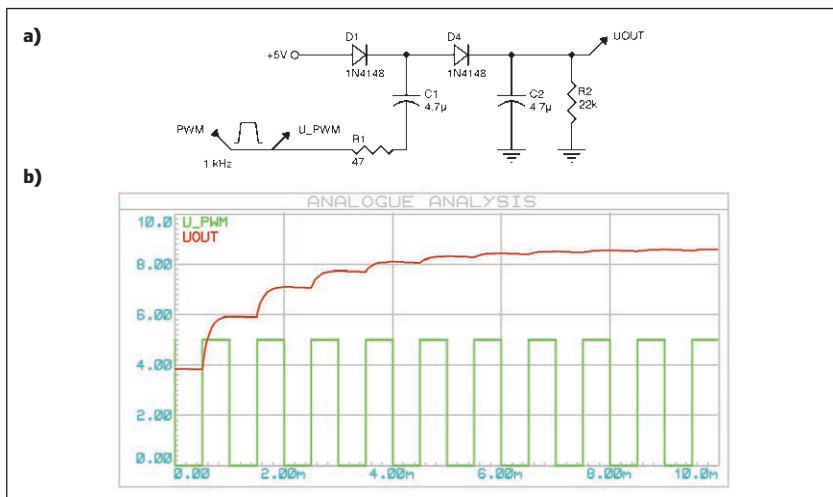


FIGURE 3

A microcontroller's PWM output can be used to easily build a charge pump. **a**—Here a voltage doubler required no more than a pair of capacitors and diodes. **b**—According to the simulation, the output voltage is close to 8.5 V on a 22-k Ω load.

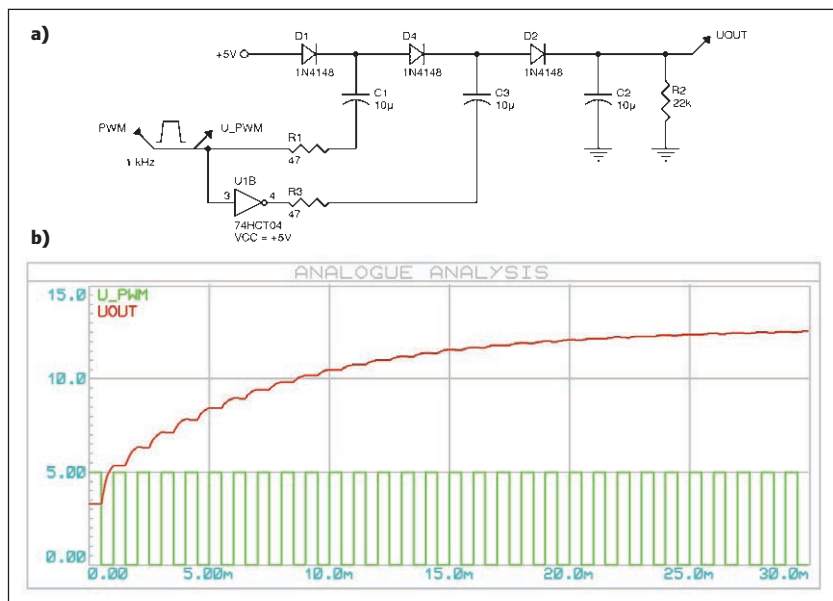


FIGURE 4

a—To build a voltage tripler, just add one more stage to the basic charge pump. **b**—Now the simulated output is close to 12 V on a 22-k Ω load.

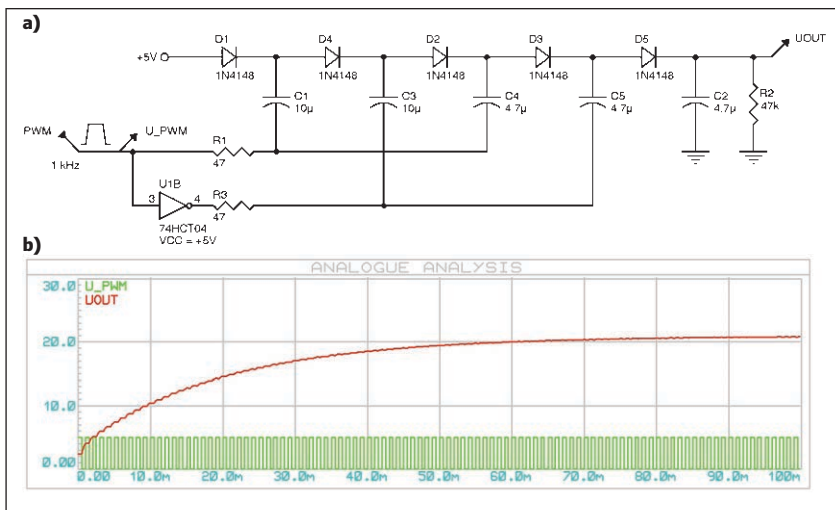


FIGURE 5

a—This schematic shows a five-stage charge pump. **b**—As shown on this simulation, this design enables the voltage to be multiplied by nearly five, providing 21 V from the 5-V source. However, multistage charge pumps have huge losses, so other solutions (e.g., a boost converter) may be adequate.

microcontroller pin. Then when the PWM signal goes high, the voltage of the positive end of C1 goes above 5 V. D1 is reverse-polarized so the current must flow through D2 and charge C2.

I used Labcenter Electronics's Proteus VSM SPICE-based simulator with a 22-k Ω resistor as a load to simulate the schematic shown in **Figure 3**. The result shows that the output voltage goes up to a stable 8.5 V. This is close

to 10 V (i.e., 2×5). The difference is mainly explained by the losses in the diodes.

By the way, you may have already used a charge pump even if you were not aware. Have you used any of Maxim Integrated's MAX232 driver/receivers? These chips integrate a charge pump step-up generator to provide RS-232 voltages. If you already have such a chip in your design, you could use it to generate above-the-rail voltages, typically around 8 V.

Can you still use a charge pump even if you require more than 8 V from your 5-V rail? Of course, but then you will need a multistage charge pump. The idea is still simple. You chain two basic charge pump circuits to triple the voltage rather than doubling it. The only trick is that the two sections should work on opposite clock phases. So you will just need some more capacitors and diodes and maybe a logic inverter gate if your microcontroller can't directly generate two inverted outputs.

Figure 4 shows you how to build a charge pump tripler. The simulation shows that this design could deliver a little more than 12 V on a 22-k Ω load, which is not so bad. If a tripler is not enough, you can add more stages. **Figure 5** an example of a quintupler based on five diodes and capacitors, providing 21 V on a 47-k Ω load, as simulated. However, adding additional stages significantly reduces the power efficiency, so charge pumps are usually limited to doublers or triplers.

TRANSFORMERS

I admit this may not look like a high-end solution, but another alternative to elevate a voltage is to use a transformer. **Figure 6** shows a basic example built around a small one-to-three-turn ratio transformer that easily provides 12 V from the 5-V source. Here, once again, the microcontroller can be used as a square-wave source through a PWM output and could drive the transformer directly, through logic buffers, or through a power stage if high power is required.

This is probably not the best idea if you just need to increase 5 to 12 V with a small power requirement, but this will definitely be an interesting option if you require 2 kV or some kilowatts! So keep this option in mind.

VOLTAGE MULTIPLIERS

What if you require a 24-V output but your transformer only has a 12-V output? Just add a voltage doubler (see **Figure 7**). According to *Wikipedia*, this type of doubler circuit was invented by Heinrich Greinacher in 1913 and was also called a "Schenkel doubler." It works a little like the capacitive charge pump I described earlier, but with an AC source. When the transformer's output is negative, D1 is conductive and D2 is locked.

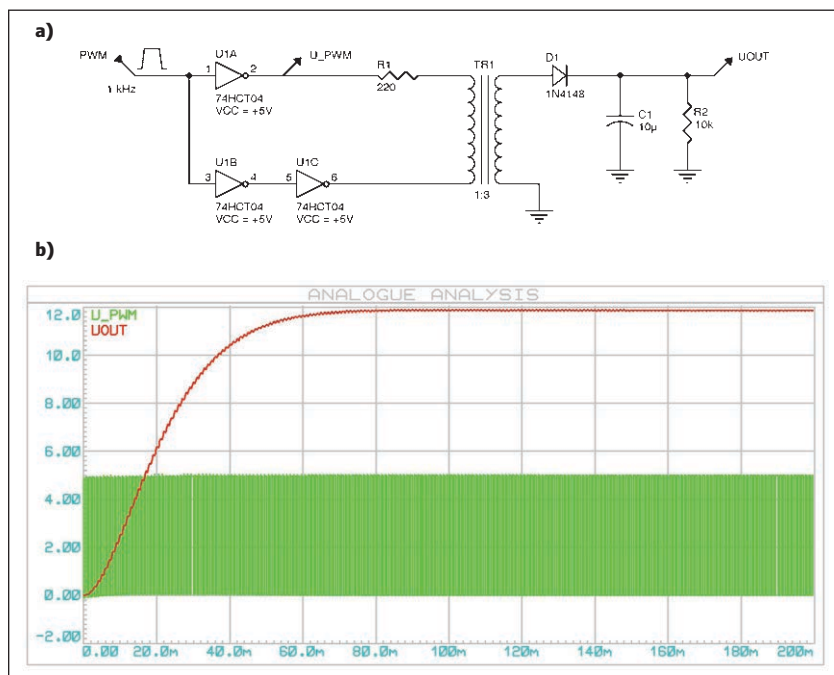


FIGURE 6

Don't forget that a transformer can also be used. **a**—This schematic shows a circuit that could be used to drive a transformer from a microcontroller PWM output through adequate drivers. **b**—The simulated output voltage is directly linked to the transformer's turn ratio.

C1 is loaded to the supply voltage through D1. During the next half period, D1 is locked and D2 is conductive. The transformer's output voltage will be added to C1's voltage and will charge C2 to twice the transformer's voltage. Of course you can also chain several Schenkel voltage multiplier cells after a transformer and increase the voltage even more.

Figure 8 shows how to use four diodes and four transistors to build a quadrupler. You could add several more stages even if, once again, the conversion efficiency become lower and lower.

A word of caution: You may think that such a circuit could be used without a transformer, replacing it with two non-isolated drivers shifted by 180°. This is true, but only if the drivers have a symmetric swing above and below 0 V (i.e., ±5 V). If not, the Schenkel cell will not work. One tricky way to solve this problem is to use a virtual ground centered at V/2, but other solutions are usually more efficient.

HANDMADE BOOST CONVERTERS

Remember the first solution I presented, which was based on a ready-made boost DC/DC converter module. Why not design such a

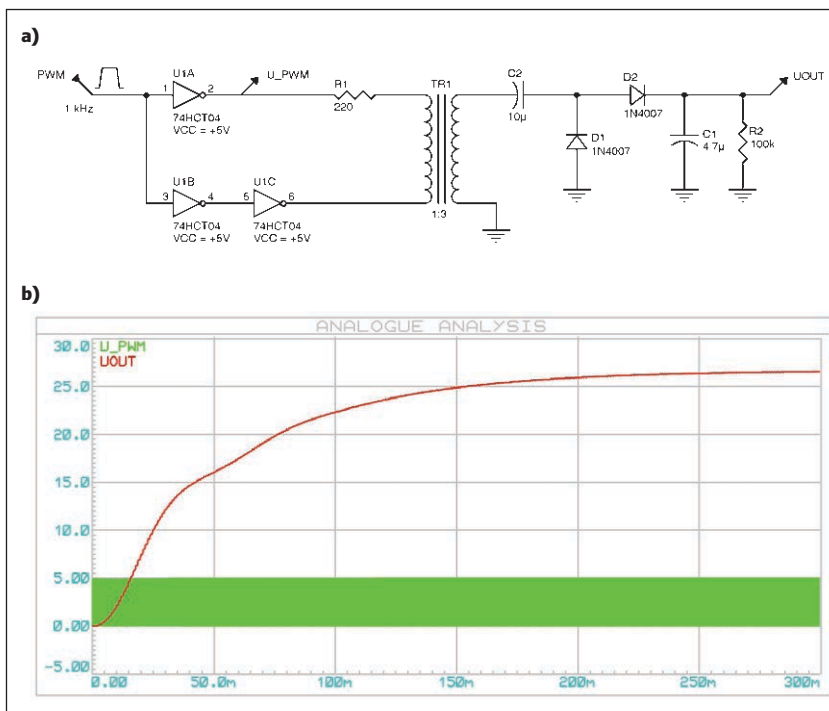


FIGURE 7
a—A voltage doubler has been added after the transformer. **b**—This simulated output voltage is doubled; compare it to Figure 6b.

COLUMNS

MSP-GANG



FlashPro430
FlashPro-CC
FlashPro2000
GangPro430
MSP-GANG
GangPro-CC

NEW

FlashPro-CM
FlashPro-LM
FlashPro-STM

USB Flash and Gang Programmers for Texas Instruments' MCUs MSP430, Chipcon CCxx, C2000, Stellaris LMxx and ST-Microelectronics MCUs - STM32xx (ARM)

USB-FPA

* can assign unique serial number
 * up to 64 USB-FPA programmers can be connected to one PC and program target devices simultaneously

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



Elprotronic
 Incorporated

www.elprotronic.com

25 YEARS Beta
 LAYOUT
 create : electronics

Create your own Solder Reflow station



NEW!

Reflow Controller:	\$ 315.⁰⁰
Large Beta Reflow Kit:	\$ 178.⁰⁰
Total Cost	\$ 493.⁰⁰

Laser Stencil for assembly **FREE** with every PCB-POOL® order: **\$ 0.⁰⁰**

www.pcb-pool.com

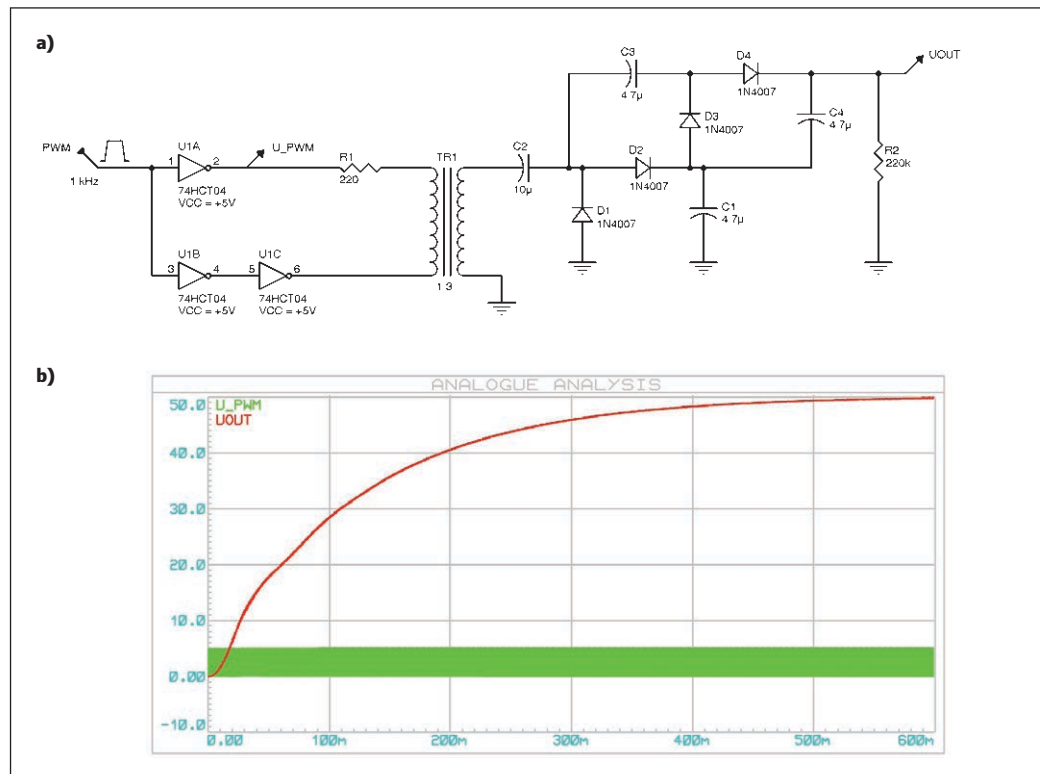
For information, please call the USA office toll free at 888-977-7443

Order online at www.beta-eSTORE.com

eSTORE
 Beta LAYOUT

FIGURE 8

a—Nothing prevents you from adding even more voltage elevation steps, such as this quad Schenkel cell multiplier. These kinds of topologies are often used to generate tens of kilovolts or more. **b**—The simulated output voltage is close to 50 V with a 12-V transformer.



converter yourself? I have already devoted an article to DC/DC converters, so I will not detail them here. (Refer to my article, "DC/DC Converter Basics," *Circuit Cellar* 239, 2010.)

A step-up boost converter requires nothing more than a PWM signal, a switching element (e.g., a transistor), an inductor, a diode, and a capacitor. **Figure 9** shows you the concept and the simulation result. Here I assumed that the microcontroller provides a 100-kHz

signal with a 30% duty cycle (more on that later). When the PWM signal is high, the Q1 transistor is conducting and the current going through the inductor L1 increases over time. When the PWM goes low, Q1 switches off. However, the current going through an inductor can't stop immediately; therefore, this current now goes through D1 and charges C1. As D1 is connected to the power supply source through D1 and L1, C1 could only be charged to a voltage higher than the supply voltage. You have a step-up converter—more specifically, a boost converter.

This solution may seem simple to implement, but some cautions apply. First, a boost converter is usually not used without a feedback loop. Why? Imagine there is no load on the output (i.e., the loading resistor shown on the simulation is removed). The energy transferred by the inductor can't be dissipated anywhere, so the voltage across the output capacitor will continue to increase until something breaks. This will usually happen with smoke or fire. This shows you that such a design will have a very unstable output voltage if the load is not stable.

If your design has a nearly constant load, this may be stable enough. You could also add another fixed charge resistor, if needed. However, you will likely need to implement a kind of control loop. Once again, this would be quite easy if you have a microcontroller on board with some spare resources. Add a resistive voltage divider on the output and

PROJECT FILES



circuitcellar.com/ccmaterials

RESOURCES

H. Darmawaskita, "DC/DC Converter Controller Using a PIC Microcontroller," Application Note 216, Microchip Technology, Inc., 2002.

J. Kronjaeger, "Basic Multiplier Circuits," *Kronjaeger.com*.

R. Lacoste, "DC/DC Converter Basics," *Circuit Cellar* 239, 2010.

—, "PID Control Without Math," *Circuit Cellar* 221, 2008.

Wikipedia, "Voltage Multiplier."

SOURCES

Proteus VSM design suite

Labcenter Electronics | www.labcenter.com

MAX232 Driver/receiver

Maxim Integrated, Inc. | www.maximintegrated.com

TME0512S Voltage DC/DC converter

Traco Electronic AG | www.tracopower.com

connect it to the microcontroller's ADC input. Then you will just have to develop firmware that will increase or decrease the PWM duty cycle when the measured output voltage is respectively above or below the desired voltage.

By the way, this is a well-documented idea. For example, you can refer to Microchip Technology's application note if you want to design such a DC/DC converter around a PIC16 (see Resources).

Another concern is that this boost converter is not protected against overcurrents through the inductor, in particular if the firmware keeps the transistor switched on too long. More precisely, there is only a limited overcurrent protection through the resistor on the transistor's base (R1). Since I used a bipolar transistor, the current through its collector will be limited by its gain times the base current. Don't replace it with a MOSFET, since switching currents in such a boost converter can become quite high.

The simulation shows that the peak current through the transistor is already ± 300 mA or so, even with 12-mA output load through the 1-k Ω resistor. You could also measure the output current through another resistor and feed it to your microcontroller's ADC input, then you could limit the overall current with some extra lines of code.

WRAPPING UP

My goal with this article was to list the different options designers can use to generate an over-the-rail power supply voltage. Using a ready-made boost converter will probably solve the problem 90% of the time, but knowing the alternatives never hurts. Moreover, this could open the road to interesting microcontroller-based projects as all these ideas could be regulated by using a firmware-based closed loop. For some reminders about that topic, go back and read my article about PID controls ("PID Control Without Math," *Circuit Cellar* 221, 2008).

Anyway, I hope that charge pumps, voltage multipliers, and boost converters are

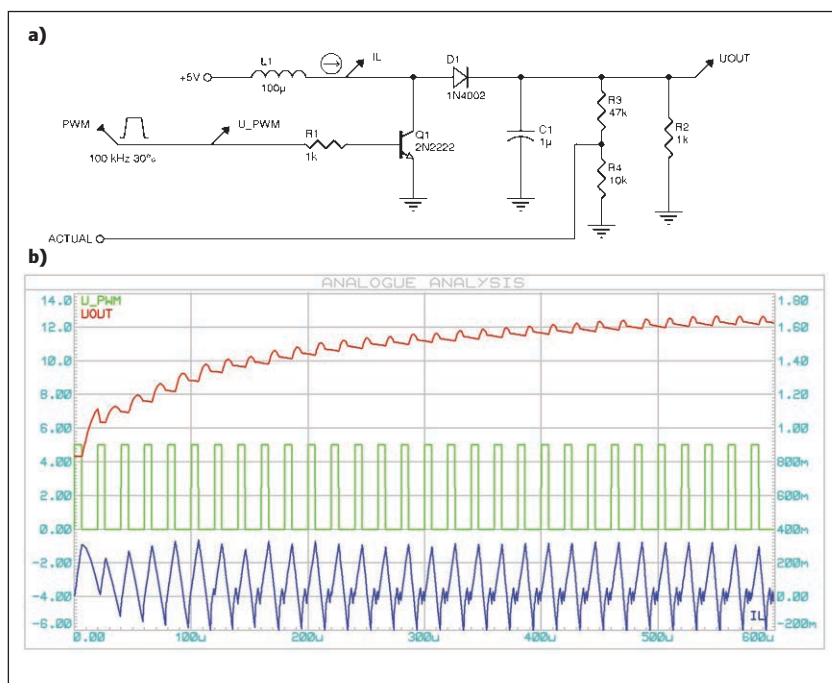



FIGURE 9

a—It is also easy to use a microcontroller's PWM output to build a boost converter stage, as shown in this schematic. However, it is wise to add a firmware control loop around this basic circuit, controlling the PWM duty cycle based on the actual voltage output. **b**—The simulated output waveform shows that 12 V is generated, as expected. The blue curve shows the current through the inductor.

no longer in the dark for you. Don't hesitate to play and experiment.

A final caution: When playing with inductors or transformers, don't forget that you are building voltage multipliers. Lethal voltages could appear even if your design is powered by a low-voltage source. So apply the basic safety rules: Always shut the power off before touching anything, keep one hand in your pocket, remember that capacitors can stay charged, and never work alone just in case the other safety rules fail! 

ABOUT THE AUTHOR

Robert Lacoste lives in France, near Paris. He has 25 years of experience in embedded systems, analog designs, and wireless telecommunications. A prize winner in more than 15 international design contests, in 2003 he started his consulting company, ALCIOM, to share his passion for innovative mixed-signal designs. His book (*Robert Lacoste's The Darker Side*) was published by Elsevier/Newnes in 2009. You can reach him at rlacoste@alciom.com. Don't forget to put "darker side" in the subject line to bypass spam filters.



FROM THE BENCH

Passive RFID Tagging (Part 2)

Front-End Analog Circuits

The first part of this article series described the history of RFID technology and covered the topic of read-only tags. This article focuses on front-end analog circuits.

By Jeff Bachiochi (US)

In Part 1 of this article series, I mentioned how radio frequency identification (RFID) may have begun with the Luftwaffe German air force. However, its first commercial use was to help farmers track their dairy cows.

In my article "Electronic Identification" (*Circuit Cellar* 24, December 1991/January 1992), I jokingly took a staged photo of my thumb with an ID chip being embedded under the nail. I was barraged with mail questioning whether this procedure was being done to all newborns. And, there was something about an invasion of privacy. This was years

before the Health Insurance Portability and Accountability Act (HIPPA) of 1996.

In my last article, I left off with a circuit description and the idea of installing surface-mount technology (SMT) devices on carrier boards to make them easier to prototype. **Photo 1** shows the SPI circuit I used for this project. Note that barrel sockets are used to mount many external components. I cut these off of an IC socket as they have 0.1" spacing between IC pins. Normally I buy all double-wipe sockets but I keep a few of these barrel pin sockets available for this purpose.

Building the front-end analog circuit on a separate PCB enables the "module" to be easily added to any microcontroller with a SPI. Using a couple of screw terminals enables the external coil to be placed at a distance from the circuitry. This makes it easy to experiment with different coils, which will affect other component values, since they can be simply removed and replaced without soldering.

OPENING DOORS

I left you with a mystery photo in Part 1 of this article series. The photo was a hint at how this project would be used. We recently added basement space to our home. We dug out underneath and propped up the house as the ground below was excavated. We added a little concrete and *voilà* we had a new space. After fulfilling a promise to my wife, Beverly, for some renovation upstairs, I can now set my sights on creating a new workspace in the basement. *Deja vu*, Steve Ciarcia's "circuit cellar" recreated.

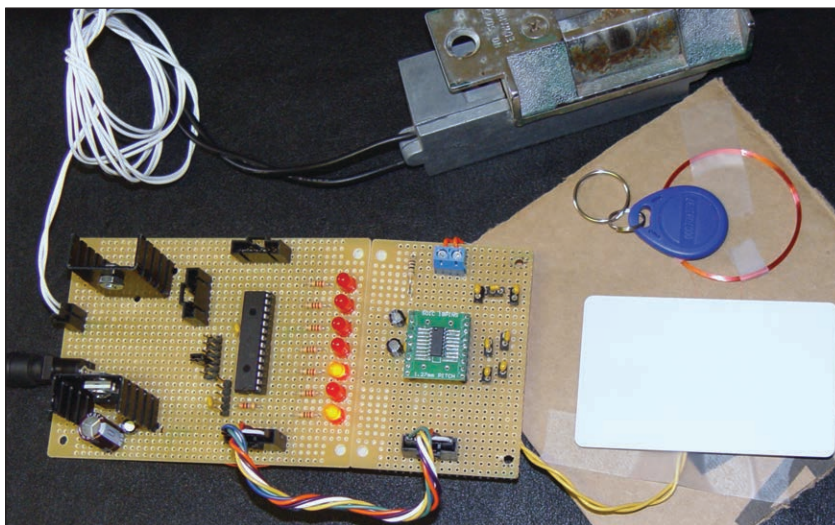


PHOTO 1

This project's front-end analog circuitry uses a SPI and receives its own PCB. I placed the EM Microelectronic EM4095 device on an SMT-to-DIP microcontroller PCB. The module easily integrates into the hand-wired prototype.

No lab would be complete without proper security. I have the perfect door lock to keep industrial spies from rifling through my secret files. I first saw these locks used at a bank to grant someone access through a locked door. This door jamb release does not have to physically unlock a door.

Normally, twisting a door handle will draw the latch into the door and out of the door jamb's strike plate, and the door can be swung open. Locking the door prevents the door handle from moving the latch and the door remains pinned in the door jamb. The door jamb's strike plate is made of metal to help prevent the door from being forced open by breaking the door jamb (especially if it's made of wood, as in a home.)

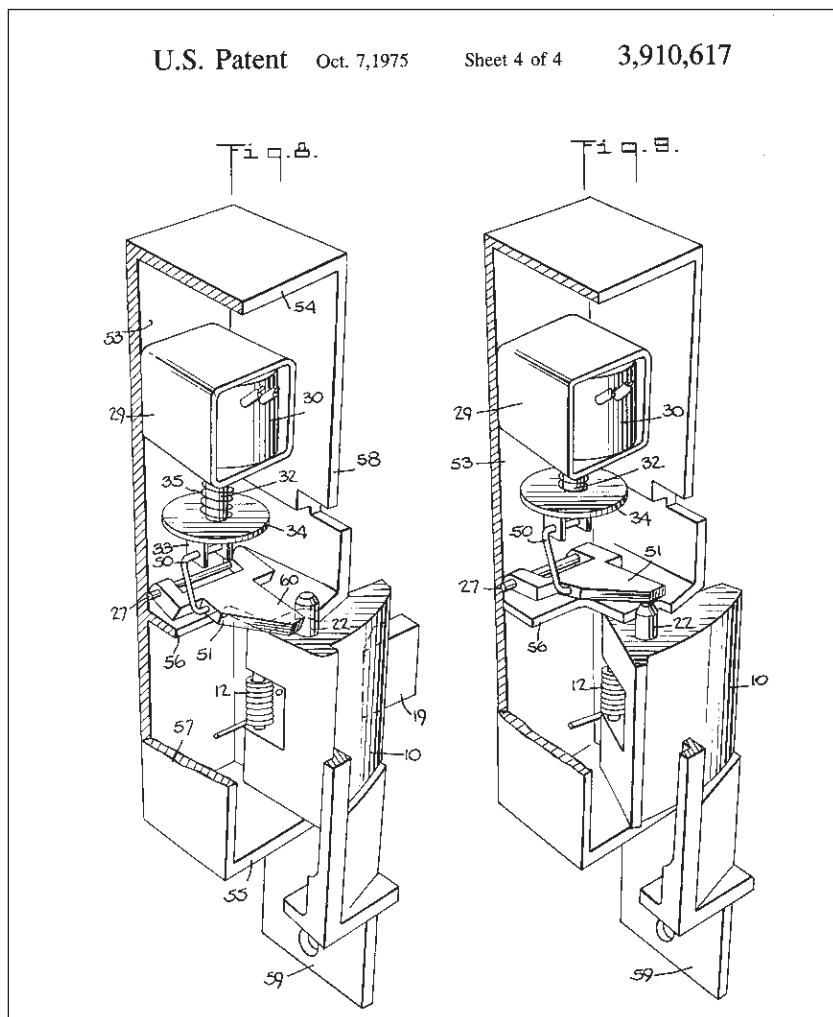
By replacing the normal strike plate with an electric strike plate, the door's latch can be released by applying an electric current. An internal electromagnet releases a catch normally preventing the latch gate from rotating and enabling the door's extended latch freedom.

Figure 1 shows the inner mechanics of this device taken from the Edwards Signaling US patent. The Edwards Signaling 151 Series Mortise-type electric door opener is energized by 24 VDC. I used a Fairchild Semiconductor RFP12N10L N-Channel logic-level power MOSFET to control this device. Using a logic-level gate makes it easy to interface directly from a microcontroller. The microcontroller could be anything with a SPI peripheral.

I used a Microchip Technology PIC18F25K22, which is a 5-V microcontroller that can run at 64 MHz from a 16-MHz internal oscillator boosted four times using the internal phase-locked loop (PLL). While a lot of I/O isn't needed, I used the extra I/O this 28-pin microcontroller offers for some configuration inputs and LED outputs. (I believe you can never have enough LEDs when you are experimenting.) I also added connections for a serial terminal I used for debugging, as I will explain shortly (see **Figure 2**).

LOOK, MA, NO BATTERIES

Most technology requires some kind of battery to operate. The whole idea of RFID—and one of growing interest today—is wireless power transmission. There are charging pads on the market that can recharge your enabled gizmo. The pad creates a radiation zone by broadcasting a signal that an antenna on your device will pick up and rectify into a charging current. RFID cards and tags use this same principle to receive enough energy to temporarily power their identification circuitry. The card modulates the load on the receiving circuitry with data. The transmitter



monitors its carrier for any modulation. Any data seen is decoded using a specific format to reveal (in this case) a 5-byte unique ID number.

Since my RFID tags use Manchester encoding and derive a bit time from a nice divisor (in this case 64) of the carrier, I used the carrier as a clock input to the SPI peripheral (SCK) and just let the SPI gather the data. Each symbol (half of a Manchester-encoded bit) is four SPI bytes long (32 clocks).

The Sleep input is brought to a Logic 0, which tells the EM Microelectronic EM4095 base station to begin transmitting the carrier. After some circuit stabilization, the carrier is output on the EM4095's CLK output. The Demod output begins indicating any modulation seen on the carrier. With no modulation (and a lack of noise), it will output a constant high to the SPI data input (SDI). A symbol will consist of four SPI bytes (32 clocks per symbol, 64 per data bit).

I used the SPI interrupt routine to collect data, which was four interrupts (bytes) per symbol. As each byte was received, I added all the 1 bits received. After four bytes, I

FIGURE 1

I based my device's locking/release mechanism on an Edwards Signaling electric strike (US Patent 3,910,617). (Image courtesy of Edwards Signaling)

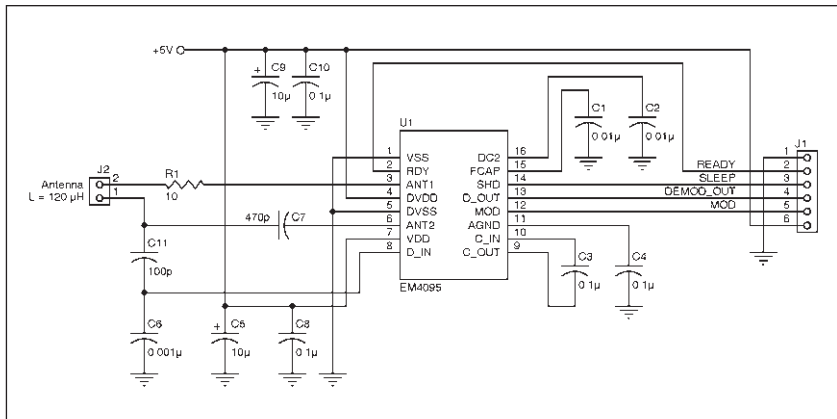


FIGURE 2

I placed the microcontroller and front-end analog circuitry on separate PCBs. This way, I can quickly exchange other options (e.g., an EM Microelectronic EM4094, which is a 13-MHz front-end device).

compared the total count to 16. If there were fewer than 16 1 bits, then the symbol would be a 0 or else the symbol would be a 1. For simplicity I saved each symbol as either a 0x00 or a 0xFF in the RFID buffer. This buffer can hold 256 symbols or 128 bytes of data. One advantage to using an in-circuit debugger is that you can stop the code execution at any point to see register values and data collected in a buffer.

I obtained memory dumps during two

attempted RFID reads. **Figure 3** shows the memory dump I took with no tag in field (TIF). **Figure 4** shows the memory dump with a TIF. With no TIF, you may expect to see all 0xFF data. In fact, you may get some sporadic 00 data. Since Manchester-encoded data must consist of 0 and 1 symbols for each bit, it is clear that “real” data must be either the symbol pattern 01 or 10 (saved in memory as FF00 or 00FF). You can never have more than two of the same symbols in a row. This makes it easy to find the division between two data bits. Wherever two of the same symbols are in a row, they must belong to separate data bits.

This is the first rule you can use to determine when to begin forming bits from symbols. The second rule is the expected format of the data sent from the tag. I thoroughly discussed both rules in Part 1 of this article series. In essence, the format consists of a preamble of nine 1 bits followed by 10 5-bit pieces of data (nibble data + row parity bit) and one 5-bit postamble (column parity + 0 bit).

SEARCHING THE HAYSTACK

If you must make real-time decisions as the data is received, comparing the data to a fixed string can be accomplished as long as you continue to receive matches. Once the strings don’t match, you need to go back and recheck for a string match after removing just one character from what has been matched so far. This requires a lot of executions, probably more than you have to spare before the next data arrives.

If you can do the search offline, after the data has been received, then you can spend whatever time is necessary to properly search the buffer. Read-only RFID tags have this advantage. Read/write tags require live comparisons, since you must correctly interrupt a tag’s transmission to get it to listen to you.

I used two separate routines (Header and Body) to search the data buffer for the expected format (see **Figure 5**). Note this search does not do any parity checking, it merely is concerned with finding a legal data string. I know a 256-byte buffer is large enough for at least one complete tag response. The format of a legal string of data begins with a header followed by the data.

Figure 5a shows a search routine that must initialize a bunch of things including Pointer1 (the buffer where the data is coming from), Pointer2 (where any matches are stored), and BackUp (the address where the match starts). The routine then goes into a loop attempting to find the header and nine 10-symbol pairs in a row (see **Figure 5b**).

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	ASCII
200	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
210	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
220	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
230	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
240	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
250	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
260	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
270	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
280	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
290	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
2A0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
2B0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
2C0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
2D0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
2E0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
2F0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
300	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
310	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
320	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
330	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
340	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
350	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
360	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

FIGURE 3

This memory dump shows the collection of data when there is no tag in field (TIF).

It then collects 55 additional symbol pairs (see **Figure 5c**). Any time you receive an error that breaks one of the two rules, you must begin the search again by resetting the variables and Pointer2 and using BackUp+1 as Pointer1's new starting address.

If the end of the buffer is reached before the data collection is completed, then the routine fails; otherwise, you may have success. "May" is the operational word here, as the 55 data bytes have not been checked for parity errors. I often use a serial connection to aid in debugging. While I did not dump the 256 bytes of raw data received, I did send successful search data to my connected terminal formatted to display the header and data as a matrix (see **Photo 2**).

To parity check the data, ensure that the total number of 1s for each 5-bit row and the number of 1s in each 11-bit column (the first four columns) are even. The final bit should be a 0. If the parity is successful, you can collect the five actual data bytes, which are the first four bits of each of the first 10 rows.

I used the serial port to display these nibbles as the ID. What's with the "Sorry, no match" message just below the ID in **Photo 2**? The answer to that can be found in the next section.

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	ASCII
200	00	00	00	FF	FF	00	00	FF	00	FF	FF	00	00	FF	FF	00
210	00	FF	FF	00	00	FF	FF	00	00	FF	FF	00	00	FF	FF	00
220	00	FF	00	FF	FF	00	FF	00	FF	00	FF	00	FF	00	FF	00
230	FF	00	FF	00	FF	00	00	FF	00	FF	00	FF	00	FF	00	FF
240	FF	00	FF	00	00	FF	00	FF	00	FF	00	FF	00	FF	00	FF
250	00	FF	00	FF	00	FF	00	FF	00	FF	00	FF	00	FF	00	FF
260	FF	00	FF	00	00	FF	00	FF	00	FF	00	FF	00	FF	00	FF
270	00	FF	00	FF	00	FF	FF	00	FF	00	00	FF	FF	00	FF	00
280	FF	00	00	FF	FF	00	00	FF	00	FF	FF	00	00	FF	FF	00
290	00	FF	FF	00	00	FF	FF	00	00	FF	FF	00	FF	00	00	FF
2A0	00	FF	00	FF	FF	00	FF	00	FF	00	FF	00	FF	00	FF	00
2B0	FF	00	FF	00	FF	00	00	FF	00	FF	00	FF	00	FF	00	FF
2C0	FF	00	FF	00	00	FF	00	FF	00	FF	00	FF	00	FF	00	FF
2D0	00	FF	00	FF	00	FF	00	FF	00	FF	00	FF	00	FF	00	FF
2E0	FF	00	FF	00	00	FF	00	FF	FF	00	FF	00	00	FF	00	FF
2F0	00	FF	00	FF	00	FF	FF	00	FF	00	00	FF	FF	00	FF	00
300	31	31	31	31	31	31	31	31	31	30	30	30	30	30	31	31	11111111 10000011
310	30	30	30	30	30	30	30	30	30	30	30	30	30	30	31	31	00000000 00000011
320	30	30	31	31	30	30	30	30	30	31	31	30	31	31	31	30	00110000 01101110
330	31	30	30	31	30	31	30	31	30	31	30	31	31	30	30	30	10010101 01011000
340	0C	00	6C	3E	25	00	00	00	00	00	00	00	00	00	00	00	..1>%...

FIGURE 4

With a tag in field (TIF), data from a tag is continually repeated as long as the transmitter is sending out the carrier. With at least one full transmission in the buffer, it can be searched for the preamble, which is the start of the transmission format.

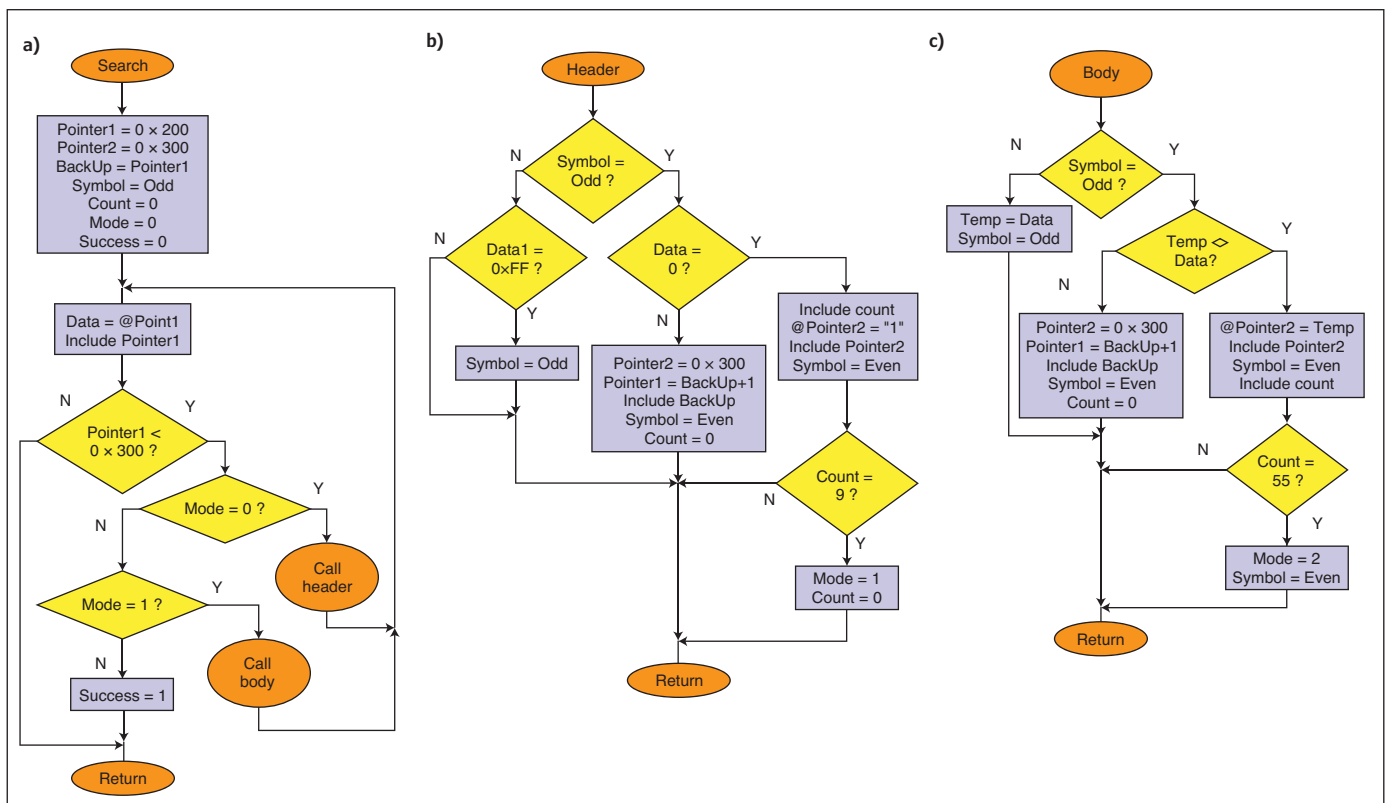


FIGURE 5
a—This flowchart demonstrates the searching algorithm used for this project. **b**—You need to identify the preamble, which is nine consecutive 1-bit symbol pairs. **c**—Once the preamble is identified, the data can be gathered from the following 55 symbol pairs.

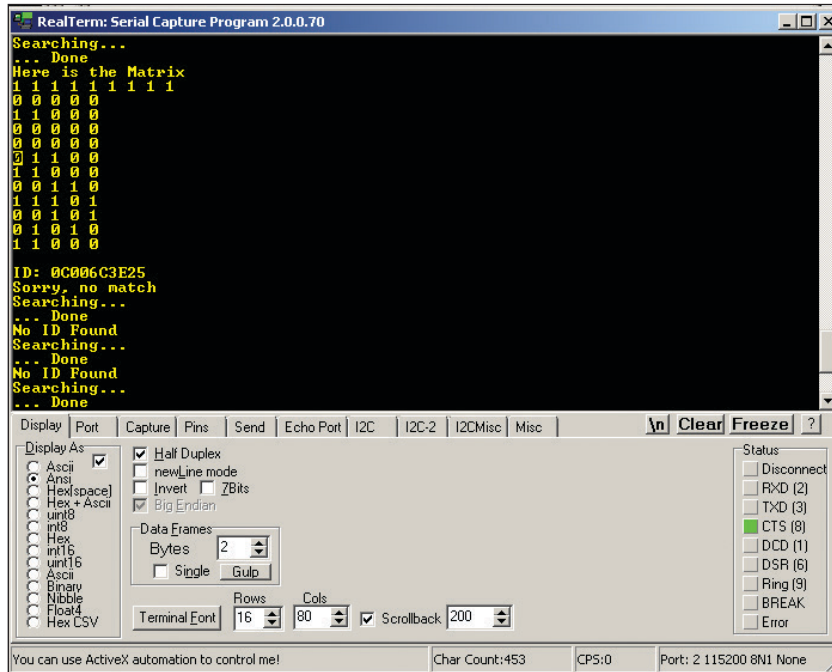


PHOTO 2

Whenever possible, I like to use LEDs as visual indicators and serial output to help debug the application. It is much easier to interpret data when it is properly formatted. Seeing the data presented in this matrix as received is much easier to understand than the memory dump.

DON'T KNOCK IT

While there is no “secret knock,” there is a secret ID! The PIC18F25K22 microcontroller has an internal EEPROM space that can be used as a semipermanent storage space. This is great for storing any data that may occasionally change yet remain even after the microcontroller’s power has been removed. The PIC18F25K22 has 256 bytes of EEPROM space. That is enough room for more than 50 separate IDs.

To simplify this project, I used a couple of jumpers to select two optional routines. One jumper adds IDs to the EEPROM and the other removes IDs from the EEPROM. When you purchase an RFID card or tag, it comes without

LED#	Indication
1	Good ID found
2	Looking for an ID
3	No ID found
4	Door unlocked
5	Add ID mode
6	Remove ID mode
7	Circuit power

TABLE 1

LEDs can be useful, as evidenced by this list of LEDs and their indications.

any documentation on its preprogrammed ID. While I could add routines to manually add and remove IDs through the serial port, it is easier to just read the tag and add or remove its ID depending on the jumper. **Table 1** shows where the LEDs come in handy.

You must insert JP1 to add an ID to the EEPROM. LED5 will illuminate, indicating you are in Add ID mode. The search process is repeated about once a second indicated by LED2. LED3 will illuminate after each search when there is no TIF.

Once a tag is brought within range and an ID is read from the device, LED1 will illuminate and the ID number is sent out the serial port. Since tags are five bytes in length, 5 can be used as an offset to the first byte of each tag’s location in the EEPROM.

A new tag’s ID is searched for in the EEPROM. If it is found, there is no reason to add it again. If it is not found, then the first empty location (five bytes of 0xFF) is used to store the new ID. The saved information’s status is also sent out the serial port. However, there is no need to monitor this, it is strictly for debugging purposes. Multiple additions of any stored IDs are avoided.

To remove an ID, the process is the same using JP2. LED6 will illuminate, indicating you are in Remove mode. Any ID cards or tags that are recognized (LED1) will be removed from the list by writing a 0xFF bytes to the five consecutive locations at the offset where the ID was found. No operation is performed if the ID is not found in the list.

One simple routine can be used to accomplish these maintenance routines. This project’s main objective is to recognize an ID tag and determine if it should grant access.

PROJECT FILES



circuitcellar.com/ccmaterials

RESOURCES

J. Bachiochi, “Passive RFID Tagging (Part 1): Read-Only Tags,” *Circuit Cellar* 286, 2014.

—, “Electronic Identification,” *Circuit Cellar* 24, December 1991/January 1992.

SOURCES

151 Series Mortise-type electric door opener
Edwards Signaling | www.edwardssignaling.com

EM4095 and EM4094 RFID Base stations
EM Microelectronic | www.emmicroelectronic.com

RFP12N10L N-Channel logic-level power MOSFET
Fairchild Semiconductor Corp. | www.fairchildsemi.com

PIC18F25K22 Microcontroller
Microchip Technology, Inc. | www.microchip.com

Figure 6 shows the match routine that does or doesn't find a matching ID in the EEPROM. This routine can also be used to find an empty entry by looking for a match of ID:FFFFFFFF. Upon leaving the routine, SearchOffset points to the beginning of the located entry.

TIME TO ENTER

Once you can successfully recognize an ID, you only need to enable the electric strike for a few seconds. This permits entry through the locked door. With a current meter on the strike, I measured 200 mA at 24 VDC for the solenoid to draw the locking plate free of the rotating latch gate. This matches the strike label, which calls out a 119- Ω coil impedance.

While the EM4095's datasheet calls out a 200-mA maximum current, with my component selection, I measured approximately 50 mA for everything but the electric strike. Mouser Electronics has a small wall wart supply that will provide 0.5 A at 24 VDC for \$8. The RFP12N10L MOSFET costs less than \$1, which is not bad for a device that can handle 12 A!

I set the RFP12N10L's enable timing for a 5-s pulse, which was sufficient time to grab the door handle and pull it free from the strike. Once the solenoid was de-energized and the closed door was again secured in the locked gate position, my laboratory secrets remained safe from prying eyes.

Of course, room security could be compromised by inadequate installation, so it is important not to allow access to the controlling circuitry from the outside. Place this safely inside the secure complex. Exposed electronics are limited to the sensing coil and status LEDs. You could restrict the exposed LEDs to one LED by programming it to both flash during search operations and remain on for the 5 s while the strike is released. A flashing LED is a great deterrent.

OTHER OPTIONS

This project uses an EM4095 125-kHz RFID analog front-end base station. EM Microelectronic also makes the EM4094, which is a similar (but not pin-compatible) 13.56-MHz RFID analog front-end base station. If you built this on its own PCB, you could use the microcontroller board and swap front ends.

This particular "specialized" front end has additional benefits. The EM4095 can also be used with read/write tags. If there is a need to store some specific information on one of these tags, the read/write tags can store multiple 29 4-byte entries as well as their preprogrammed 4-byte serial and ID numbers. It's not an enormous amount of data, but it can be "password" protected if required. You can experiment with read/write

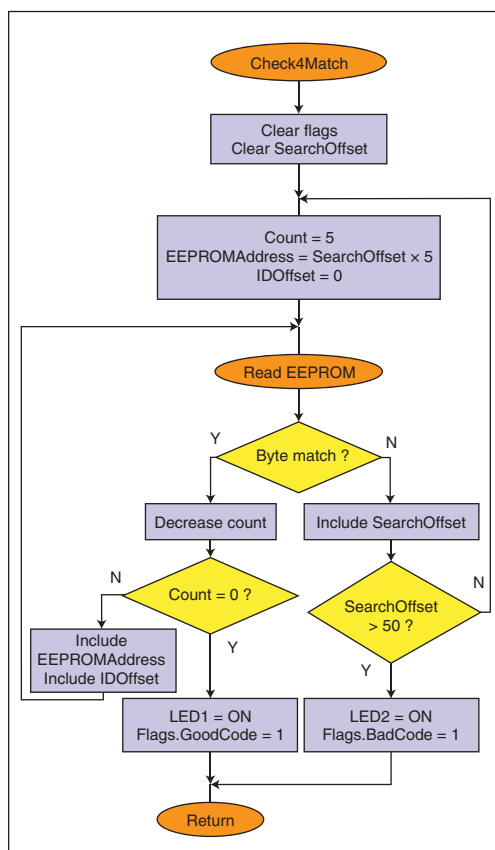



FIGURE 6

Once the ID can be read from a tag in field (TIF), you must determine whether the ID is on the guest list. The internal EEPROM can hold many "friendly" IDs. This routine identifies the presence or absence of an ID in the EEPROM and releases the door, if appropriate. EEPROM maintenance (i.e., the adding and removing of IDs), also uses this routine, which makes it central to the application.

tags using the same circuitry presented here.

If you have purchased a new car in the last few years, you are probably familiar with (or been annoyed by) the tire pressure monitoring system (TPMS). This is an RFID device that takes its data from a pressure sensor as opposed to pre-stored ID bytes. Readers located in each wheel well receive the data and illuminate an idiot light if any tire's pressure is out of tolerance (usually a 25% loss of the recommended pressure).

In most cases credit and debit cards do not carry any balance information on their magnetic strips. While these strips can carry some additional information, it is mostly the same stuff that you can read from the card (e.g., issuer, card number, expiration date, security digits, etc.). We are beginning to see some RFID technology embedded into these cards. Imagine carrying all of your personal and financial information on an RFID card or a chip embedded in your body. Is this *deja vu*, or is it time to revamp the banking system? Can you say "bitcoin?" 

ABOUT THE AUTHOR

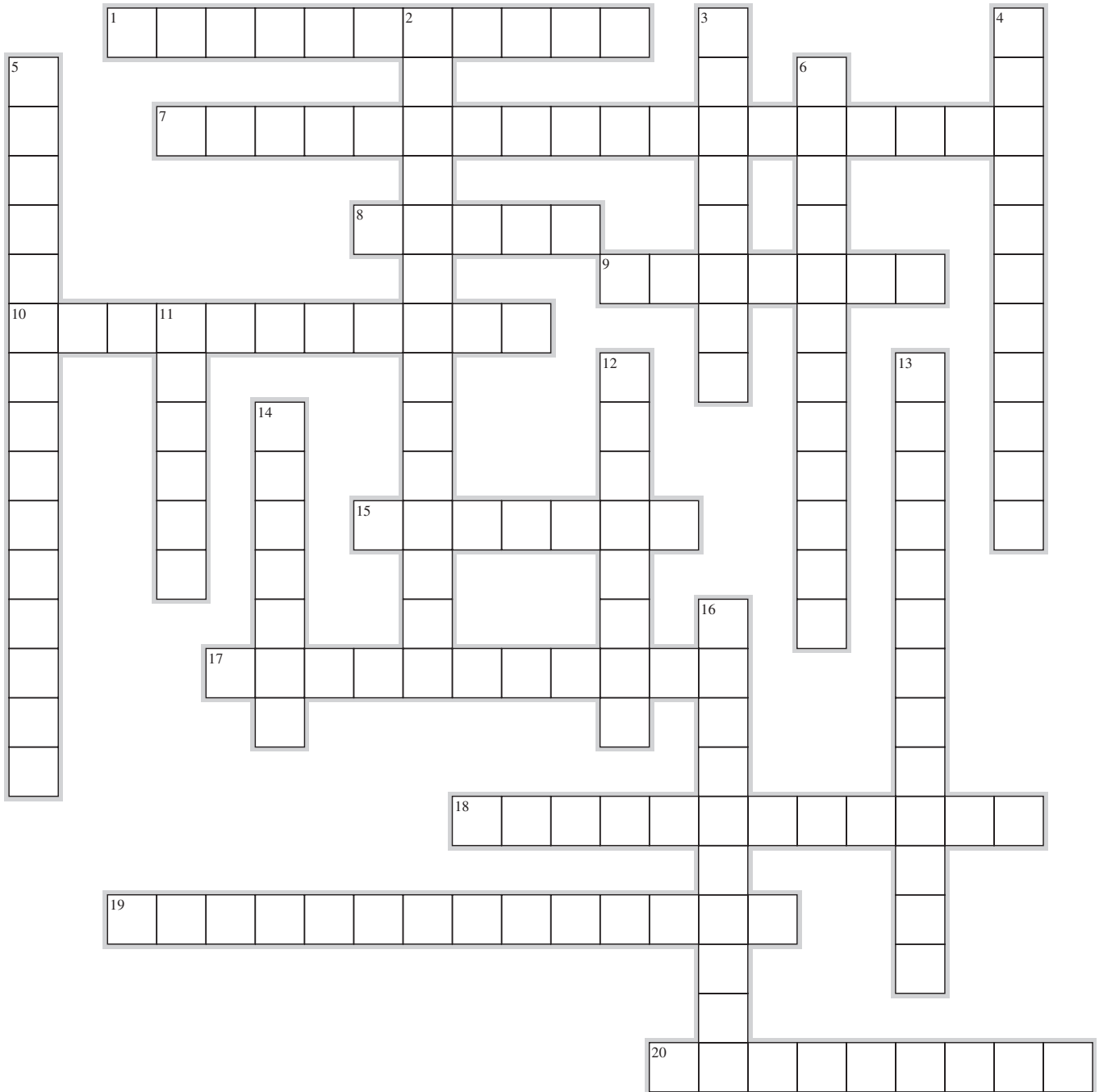
Jeff Bachiochi (pronounced BAH-key-AH-key) has been writing for *Circuit Cellar* since 1988. His background includes product design and manufacturing. You can reach him at jeff.bachiochi@imagine-thatnow.com or at www.imaginetatnow.com.



CROSSWORD

JUNE 2014

The answers will be available at circuitcellar.com/crossword.



ACROSS

1. Signal processing technique
7. Converts a signal to an intermediate frequency [two words]
8. A high-resolution PC based on Motorola's 6800 microprocessor family
9. Creates Russian nesting doll-type packets to improve a TCP/IP network's performance
10. A thin circuit board designed for one specific application [two words]
15. Tests for coding and security errors
17. Nonvolatile RAM [two words]
18. Restricts access to Web content and services [two words]
19. A PCB interface that goes between a parallel LCD and a microcontroller [two words]
20. Open-source, Linux-based OS

DOWN

2. Determines an AC wave's voltage [three words]
3. IBM's active matrix LCD
4. Happens when a fatal error is detected [two words]
5. Requires a state transition at the end of every data bit [two words]
6. Ability to be polarized
11. Helps create realistic 3-D graphics
12. An audio process that combines signals to create a comb filter effect
13. Used for circuit design experimentation
14. Five of these equal approximately $4.5 \times 10^{12} \Omega$
16. Utilizes human speech to code

What's your EQ? The answers are posted at www.circuitcellar.com/category/test-your-eq/. You can contact the quizmasters at eq@circuitcellar.com.

ANSWER 1

Pretty much any large chunk of combinatorial logic can be pipelined to reduce the clock period. This enables it to produce more results in a given amount of time at the expense of increasing the latency for any particular result.

Divider logic is easy to pipeline and the number of pipeline stages you can use is fairly arbitrary. You could insert a pipeline register after each subtract-MUX pair, or you may choose to do two or more subtract-MUX stages per pipeline register. You could even go so far as to pipeline the subtracts and the MUXes separately (or even pipeline *within* each subtract) to get the fastest possible clock speed, but this would be rather extreme.

The more pipeline registers you use, the shorter the critical path (and the clock period) can be, but you use more resources (the registers). Also, the overall latency goes up, since you need to account for the setup and propagation times of the pipeline registers in the clock period (in addition to the subtract-MUX logic delays). This gets multiplied by the number of pipeline stages to compute the total latency.

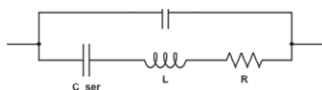
ANSWER 2

If you don't need the level of performance provided by a pipelined divider, you can compute the quotient serially, one bit at a time. You would just need one subtractor and one multiplexer, along with registers to hold the input values, the quotient bits, and the intermediate result.

You could potentially use additional subtract-MUX stages to compute more than one bit per clock period. This gives you the flexibility to trade off space and time as needed for a particular application.

ANSWER 3

The equivalent circuit for a quartz crystal is something like this:



The components across the bottom represent the crystal's mechanical resonance, while the capacitor at the top represents the capacitance of the electrodes and holder. Typical values are:

- C_{SER} : 10s of fF (yes, femtofarads, 10^{-15} F)
- L : 10 s of mH
- R : 10 s of Ω
- C_{PAR} : 10 s of pF

The crystal has a series-resonant frequency based on just C_{SER} and L . It has relatively low impedance (basically just R) at this frequency. It also has a parallel-resonant (sometimes called "antiresonant") frequency when you consider the entire loop, including C_{PAR} .

Since C_{SER} and C_{PAR} are essentially in series, together they have a slightly lower capacitance than C_{SER} alone, so the parallel-resonant frequency is slightly higher. The crystal's impedance is very high at this frequency.

But at frequencies much higher than either of the resonant frequencies, you can see that the impedance of C_{PAR} alone dominates and it continues decreasing with increasing frequency. This reduces the crystal lattice filter to a simple capacitive divider, which passes high frequencies with little attenuation.

ANSWER 4

First, calculate the crystal's equivalent inductance, based on the

TEST YOUR EQ

Contributed by David Tweed

series-resonant frequency:

$$L = \frac{1}{(2\pi f_{SER})^2 C} = 9.3535377 \text{ mH}$$

Next, calculate the capacitance required to resonate with that inductance at the parallel-resonant frequency:

$$C_X = \frac{1}{(2\pi f_{PAR})^2 L} = 26.9852 \text{ fF}$$

Finally, calculate the value of C_{PAR} required to give that value of capacitance when in series with C_{SER} :

$$\frac{1}{C_X} = \frac{1}{C_{SER}} + \frac{1}{C_{PAR}}$$

$$C_{PAR} = 6.37 \text{ pF}$$

Note that all three equations can be combined into one, and this reduces to:

$$C_{PAR} = \frac{C_{SER}}{\left(\frac{f_{PAR}}{f_{SER}}\right)^2 - 1}$$

CONNECT WITH

Circuit Cellar

For people who are passionate about hardware and software design, embedded development, and computer applications. Awesome projects, industry news, trivia challenges, and more!

Connect.

Follow us on Twitter. Like us on Facebook.

circuitcellar.com

@circuitcellar
@editor_cc

circuitcellar

CC SHOP

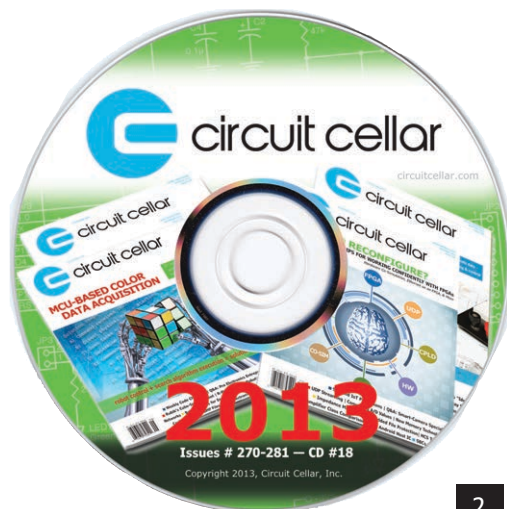


1

1 CC VAULT

CC Vault is a pocket-sized USB that comes fully loaded with every issue of *Circuit Cellar* magazine! This comprehensive archive provides an unparalleled amount of embedded hardware and software design tips, schematics, and source code. CC Vault contains all the trade secrets you need to become a better, more educated electronics engineer!

Item #: CCVAULT



2

2 CC 2013 CD

2013 was an exciting year for electronics engineers! The continued success of open-source solutions, Internet of Things (IoT) revolutions, and green-energy consciousness has changed the face of embedded design indefinitely. In *Circuit Cellar's* 2013 archive CD, you can find all of these hot topics and gain insight into how experts, as well as your peers, are putting the newest technologies to the test. You'll have access to all articles, schematics, and source code published from January to December 2013.

Item #: CD-018-CC2013

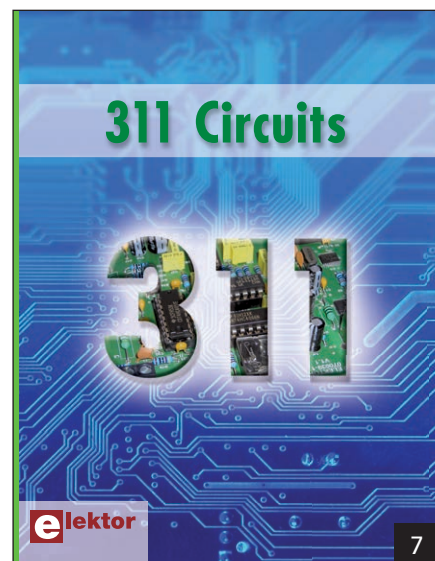


3

3 MICROPROCESSOR DESIGN USING VERILOG HDL

After years of experience, Monte Dalrymple has compiled his knowledge of designing embedded architecture and microprocessors into one comprehensive guide for electronics engineers. *Microprocessor Design Using Verilog HDL* provides you with microarchitecture, writing in Verilog, Verilog HDL review, and coding style that enables you to depict, simulate, and synthesize an electronic design on your own.

Author: Monte Dalrymple
Item #: CC-BK-9780963013354



4 311 CIRCUITS

An immense source of inspiration for all electronics enthusiasts and professionals, *311 Circuits* deserves a place on your bookshelf! This book includes tips in all areas of electronics: audio and video, computers and microcontrollers, power supplies and batteries, test and measurement, and more. The 12th book in Elektor's celebrated 300 series, it presents complete solutions for numerous problems and distinct starting points for your DIY projects.

Author: Elektor
Item #: BK-ELNL-978-1-907920-08-0

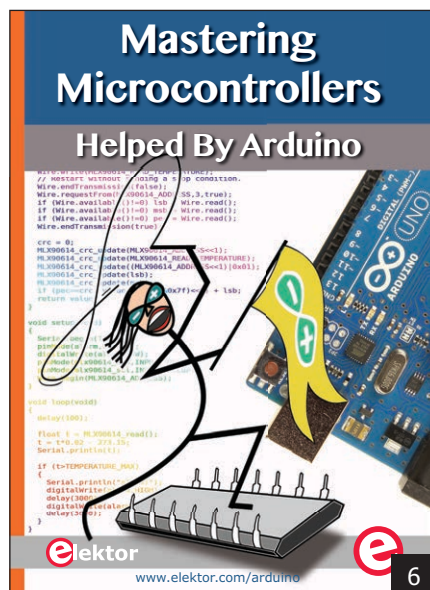


5 ANDROID APPS: PROGRAMMING STEP-BY-STEP

Many smartphones and tablet computers are powered by an Android OS. These portable devices' speed and computing power enable them to run applications that would have previously required a desktop PC or custom-designed hardware. *Android Apps* introduces you to the programming required to design apps for Android devices. Operating the Android system is explained step-by-step to show how personal applications can be easily programmed.

Author: Stefan Schwark

Item #: BK-ELNL-978-1-907920-15-8



6 MASTERING MICROCONTROLLERS: HELPED BY ARDUINO

Arduino boards have become hugely successful. They are simple to use and inexpensive. *Mastering Microcontrollers* will teach you how to program microcontrollers and help you turn theory into practice using an Arduino programming environment. Become a master today!

Author: Clemens Valens

Item #: BK-ELNL-978-1-907920-23-3



7 ADUC841 MICRO- CONTROLLER DESIGN MANUAL

This book presents a comprehensive guide to designing and programming with the Analog Devices ADuC841 microcontroller and other microcontrollers in the 8051 family. It includes a set of introductory labs that detail how to use these microcontrollers' most standard features, and includes a set of more advanced labs, many of which make use of features available only on the ADuC841 microcontroller.

The more advanced labs include several projects that introduce you to ADCs, DACs, and their applications. Other projects demonstrate some of the many ways you can use a microcontroller to solve practical problems. The Keil μ Vision4 IDE is introduced early on, and it is used throughout the book. This book is perfect for a university classroom setting or for independent study.

Author: Shlomo Engelberg

Item #: CC-BK-9780963013347



8 LINEAR AUDIO SERIES

Linear Audio is a series of bookzines full of unique content you won't find anywhere else. Each book offers everything from tutorials to circuit and system design, to test reports and book reviews. Why wait? Read, learn, and do it yourself! See website for all seven editions.

Author: Jan Didden

Further information and ordering

www.cc-webshop.com

CONTACT US:

Circuit Cellar, Inc.

111 Founders Plaza, Suite 300

East Hartford, CT 06108

USA

Phone: 860.289.0800

Fax: 860.461.0450

E-mail: custservice@circuitcellar.com

IDEA BOX

the directory of PRODUCTS & SERVICES

AD FORMAT:

Advertisers must furnish digital files that meet our specifications (circuitcellar.com/mediakit).

ALL TEXT AND OTHER ELEMENTS MUST FIT WITHIN A 2" x 3" FORMAT. E-mail adcopy@circuitcellar.com with your file.

For current rates, deadlines, and more information contact Peter Wostrel at 978.281.7708 or circuitcellar@smmmarketing.us.

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

"Your reliable outsourcing partner"

LOW cost with HIGH quality PCB, PCBA and More

INSTANT QUOTE AT:
www.myropcb.com

OR CALL:
1-888-PCB-MYRO



CCS C Compiler for Microchip PIC MCUs

NEW Project Watch

Detailed log of the working project
Daily Activity • Time Log • Comments



www.ccsinfo.com/CCpw
sales@ccsinfo.com • 262-522-6500 x 35

Giga-snaP BGA Sockets and Adapters

Giga-snaP BGA SMT Adapters allow affordable socketing

- Ultra low insertion force
- Up to 2000 pins
- GHz bandwidth
- 0.5 to 1.27mm pitch
- Same CTE as PCB
- Industry's toughest socket
- Maximum solderability



Ironwood ELECTRONICS 1-800-404-0204
www.ironwoodelectronics.com

ALL ELECTRONICS CORPORATION

Electronic and Electro-mechanical Devices, Parts and Supplies. Many unique items.

We have what you need for your next project.



www.allelectronics.com
Free 96 page catalog 1-800-826-5432

PIC-SERVO MOTION CONTROL

MOTION CONTROLLERS FOR BRUSH, BRUSHLESS AND STEPPER MOTORS.

- controller chips
- controller boards

www.picservo.com
JEFFREY KERR, LLC

FPGA Boards from JAPAN **HUMAN DATA**

SAVING COST=TIME with readily available FPGA boards

ACM Series [Altera]
You can select from Cyclone V, Arria II, MAX II and other many FPGA boards

XCM Series [Xilinx]
You can select from Kintex-7, Spartan-6, Virtex-5 and other many FPGA boards

- Basic and simple features, single power supply operation
- Over 100 varieties of FPGA/CPLD boards are available

PLCC68 Series

- Designed for 68-pin PLCC socket
- Very small size (25.3 x 25.3 [mm])
- 50 I/Os (External clock inputs are available)
- 3.3V single power supply operation

We can provide you a wide range of solutions. See all our products at : **www.hdl.co.jp/CC/**



BECOME
a member of
Circuit Cellar!


Project articles and design applications
.....
Embedded industry news and announcements
.....
12 issues annually
.....
CC.Post e-newsletter

MCU-BASED COLOR DATA ACQUISITION

circuitcellar.com/
subscriptions

LISTEN TO YOUR MACHINES

Ethernet PLCs for OEMs



FMD88-10 and FMD1616-10

Integrated Features :

- ETHERNET / Modbus TCP/IP
- 16 or 32 digital I/Os
- 10 analog I/Os
- RS232 and RS485
- LCD Display Port
- I/O Expansion Port
- Ladder + BASIC Programming

\$229 and \$295
before OEM Qty Discount

tel : 1 877 TRI-PLCS
web : www.triplc.com/cci.htm

TRI TRIANGLE RESEARCH INTERNATIONAL

MaxBotix[®] inc.

High Performance Ultrasonic Rangefinders

Save 10%
Online Web-Order Code:
CIRCUIT140630
Valid thru July 1, 2014

HRXL-MaxSonar[®]-WR™

- High noise tolerance
- IP67 rated
- 1 mm resolution
- Multi-Sensor operation
- Calibrated beam pattern
- Starting at \$109.95



XL-MaxSonar[®]-EZ™

- Great for UAV's and robotics
- Incredible noise immunity
- Small in size
- 1cm resolution
- Automatic calibration
- Starting at \$39.95



Phone: 218-454-0766 Email: sales@maxbotix.com
www.maxbotix.com

4.3", 5", 7", 10.1" VESA or Wall-Mount

Techsol Engineering



Resistive or Capacitive

Touch-Screen Computer

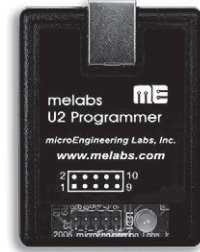


www.techsoleng.com

microEngineering Labs, Inc.

www.melabs.com 888-316-1753

Programmers for Microchip PIC[®] Microcontrollers



Starting at \$79.95

PC-Tethered USB Model (shown):

- Standalone software
- Command-line operation
- Hide GUI for automated use
- Override configuration with drop-downs

Stand-Alone Field Programmer:

- Power from target device or adapter
- Program file stored on SD-CARD
- Programming options stored in file
- Single-button operation

Program in-circuit or use adapters for unmounted chips.
Zero-Insertion-Force Adapters available for DIP, SOIC, SSOP, TQFP, and more.

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

Bluetooth Low Energy Changes the “Wireless Landscape”

By Eric VanWyk



Eric VanWyk is co-founder of Mooshim Engineering and an adjunct instructor at Franklin W. Olin College of Engineering in Needham, MA, where he earned his BSc in Electrical and Computer Engineering in 2007. His background is in educational robotics, short-range wireless, and medical device development. Eric and his business partner, James Whong, have joined the rapidly growing number of innovators developing hardware and sensor add-ons that take advantage of Bluetooth Low Energy (BLE) 4.0 in today's mobile devices. Their crowdfunded Mooshimeter is a multichannel circuit testing meter that uses a smartphone or tablet, via BLE, as a wireless, high-resolution graphical display. For more information, visit <http://mooshim.com>.



Mooshim Engineering's Mooshimeter displays a car startup transient.

In 2010, the Bluetooth Special Interest Group (SIG) took Nokia's existing Wibree standard and renamed it Bluetooth Low Energy (BLE). In doing so, it combined the latest in a series of evolutionary engineering improvements with brute-force market pressure to change the wireless landscape.

Adding BLE to the Bluetooth 4.0 specification has spurred rapid adoption. In fact, the SIG predicts that 90% of Bluetooth-enabled smartphones will support BLE by 2018. Before this wide adoption, a Wibree-based product had to include both sides of the radio link. Now a BLE-based device can ship with the assumption that the customer already owns the receiving half. This enables system architects to consider the user interface (UI) to be a software problem, not a hardware one. Hardware UIs are expensive and their power requirements are many orders of magnitude higher. BLE-based design can cut total product costs by more than half and increase usability by leveraging the customer's smartphone. This provides a high-resolution screen, an already familiar user experience, and an Internet connection essentially for free.

Wibree's main technical value proposition is its extremely small power draw. Our company, Mooshim Engineering, offers the Mooshimeter, a wireless multimeter and data logger that uses your smartphone as a display. The transceiver we use for the Mooshimeter consumes a little less than 100 μ W average draw to both send broadcast announcements every few seconds and listen for wake-up requests. This is roughly 10 to 100 times more power than a quartz watch, but 10 to 100 times less power than the watch's backlight. Like the wristwatch, this draw is extremely peaky and depends heavily on usage. Products that only need to transmit can pull as little as 155 μ J per announcement. This provides more than a year of standby time.

Using 100- μ W average draw as a starting point and assuming perfect power conversion, power could be provided by 2 to 4 mg per day of storage with a rechargeable lithium-ion battery; 1 to 3 g per day of storage with a supercapacitor; 10 mm^2 of solar cells placed in a good spot outdoors; 5 cm^2 of skin contact using thermal harvesters (e.g., a narrow but secure wristband); vibration harvesters, either on our limbs or in heavy industrial settings; or -10 dBm of wireless power transfer. An alkaline AA battery could ideally provide four

years of service, although its self-discharge is more than 10% of the energy budget.


These power levels enable devices to have a high level of energy independence and become truly wireless—no data wire and no power wire. Thus architects can explore new relationships among devices, their environments, and their users. Connectors don't compromise environmental seals, and frequent recharging doesn't compromise the user's experience.

The 100- μ W budget assumes the device just periodically announces its existence (as with wireless tethers and remote wake-up). But in uses that require more interesting payloads, the value proposition may be that the wireless link can fade into the noise of the energy budget. Remoting the user interface can also save energy, as even a dim indicator LED draws a milliwatt.

BLE is gaining its heaviest traction in electronic wearables, where users are likely to have BLE-enabled smartphones and a willingness to try new technologies. Fitness aids are enjoying early success because their sensor payloads are relatively low power and they address a large user base.

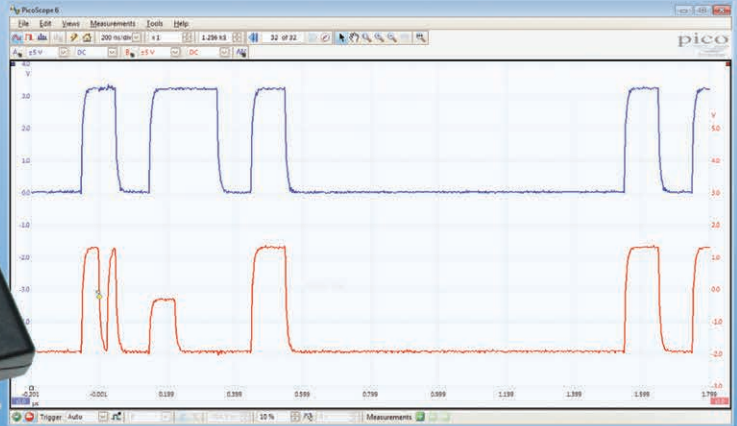
Medical wearables will take longer because of regulatory concerns and the user base. Diabetics may carry several screens with them, and often these devices will use proprietary radio protocols. Moving to a standard protocol could reduce the carry burden and provide a more secure data link. Standardization improves security through the principle of “given enough eyeballs, all bugs are shallow.”

Home appliances may be third-wave BLE adopters. The power draw is irrelevant here. A high-efficiency transformer wastes 10,000 times more power than the radio uses. It likely won't eliminate the need for a hardware user interface either. Who wants to load an app to microwave their dinner? The convincing use case is to provide powerful diagnostic and monitoring capabilities. A refrigerator can tell a user's phone when it needs a new filter. Washing machines can push notifications. Smoke detectors will proactively demand replacement batteries.

Until 2010, Wibree and its competitors offered incrementally improved energy independence. But BLE's rapid market growth offers an inexpensive and unobtrusive way for system architects to provide new and compelling user experiences. 

PicoScope[®] 2200A Series

Like a benchtop oscilloscope, only smaller and better



- Up to 200 MHz bandwidth
- 1 GS/s sampling
- Advanced digital triggers
- AWG
- Serial decoding
- USB powered
- Ultra compact design

PicoScope[®] 5000 Series

Flexible resolution oscilloscopes

- Resolution from 8 to 16 bits
- 200 MHz analog bandwidth
- 1 GS real-time sampling
- 512 MS buffer memory
- 200 MS/s AWG



8 · 12 · 14 · 16
FLEXIBLE RESOLUTION

PicoScope	PicoScope 5442	PicoScope 5443	PicoScope 5444
Channels	4	4	4
Bandwidth	All modes: 60 MHz	8 to 15-bit modes: 100 MHz 16-bit mode: 60 MHz	8 to 15-bit modes: 200 MHz 16-bit mode: 60 MHz
Sampling rate - real time	1 GS/s (8-bit mode)		
Buffer memory (8-bit) *	32 MS	128 MS	512 MS
Buffer memory (≥ 12-bit)*	16 MS	64 MS	256 MS
Resolution (enhanced)**	8 bits, 12 bits, 14 bits, 15 bits, 16 bits Hardware resolution + 4 bits		
Signal Generator	Function generator or AWG		

2 Channel models also available * Shared between active channels ** Maximum resolution is limited on the lowest voltage ranges; ±10 mV = 8 bits • ±20 mV = 12 bits. All other ranges can use full resolution.

ALL MODELS INCLUDE PROBES, FULL SOFTWARE AND 5 YEAR WARRANTY. SOFTWARE INCLUDES MEASUREMENTS, SPECTRUM ANALYZER, SDK, ADVANCED TRIGGERS, COLOR PERSISTENCE, SERIAL DECODING (CAN, LIN, RS232, I²C, I²S, FLEXRAY, SPI), MASKS, MATH CHANNELS, ALL AS STANDARD, WITH FREE UPDATES.

www.picotech.com/pco520

PRINTED CIRCUIT BOARDS & ASSEMBLY

THERE
HAS TO BE A
CATCH!

**FREE
PARTS?**

No Catch. This is for Real
No Quantity Restriction

**LET US FABRICATE AND ASSEMBLE YOUR BOARDS,
AND WE WILL PAY UP TO \$500 FOR THE PARTS.**

FIRST TIME CUSTOMERS ONLY



imagineering inc.

www.PCBnet.com • (847) 806-0003 • sales@pcbnet.com