circuitcellar.com
WIRELESS COMMUNICATIONS
FEBRUARY 2014
ISSUE 283

circuit cellar

INTERNET OF THINGS REVOLUTION

Engineer Complex MCU-Based M2M Networks

- 2013 Code Challenge Winners | Q&A: Open-Source Evangelist
- Wireless Water Alarm | Audio System Update | Home Energy Control
- FPGA Design in C | Robust Flash Memory | RF Communications | World Maker Faire | Dielectric Absorption
- Inkjet-Printed Circuitry

Serial-to-Ethernet Solutions with SSL & SSH Security

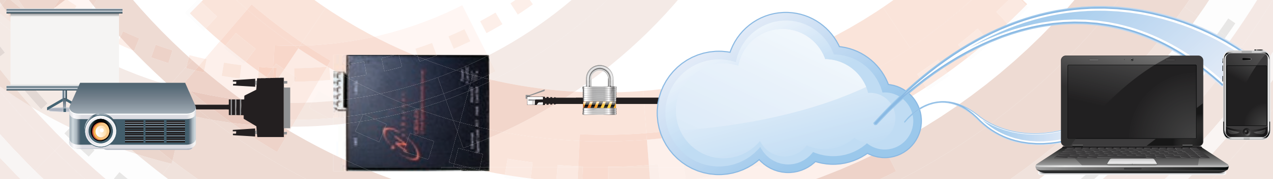
SB70 LC
2-Port 3.3V TTL serial device support

SB700 EX
2-Port RS-232 & RS-422/485 serial device support

CB34 EX
2-Port RS-232 & RS-422/485 serial device support

PK70 EX 232/485/MMS
4-Port RS-232 and/or RS-485/422 serial device support

Access serial ports over SSL & SSH protected Ethernet!



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

The method:
Connect serial port to
a serial to Ethernet device server

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

NetBurners Secure Serial to Ethernet Servers network-enable serial devices out of the box - no programming or development is required. The hardware is pre-programmed to convert your serial data to Ethernet, enabling secure communication with the serial device over a network or the Internet.

Part No. SB70LC-100IR........$49 (qty. 100)
Part No. SB700EX-100IR........$149 (qty. 100)
Part No. CB34-EX-100IR........$179 (qty. 100)
Part No. PK70EX-232CR........$285 (qty. 100)
Part No. PK70EX-485CR........$288 (qty. 100)
Part No. PK70EX-MMSCR........$346 (qty. 100)

Need a custom solution?

NetBurner Development Kits are available to customize any aspect of operation including web pages, data filtering, or custom network applications.

For additional information please visit
http://www.netburner.com/kits

NetBurner
Networking in One Day!

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

freescale™
Alliance Member

Issue 283 February 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

EDITOR'S LETTER

PLACES FOR THE IoT INSIDE YOUR HOME

It's estimated that by the year 2020, more than 30 billion devices worldwide will be wirelessly connected to the IoT. While the IoT has massive implications for government and industry, individual DIYers have long recognized how projects that enable wireless communication between everyday devices can solve or avert big problems for homeowners.

Our Wireless Communications issue features two such projects, including Raul Alvarez Torrico's Home Energy Gateway, which enables users to remotely monitor energy consumption and control household devices (e.g., lights and appliances). A Digilent chipKIT Max32-based embedded gateway/web server communicates with a single smart power meter and several smart plugs in a home area wireless network (p. 38).

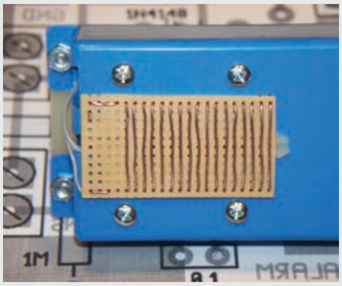
"The user sees a web interface containing the controls to turn on/off the smart plugs and sees the monitored power consumption data that comes from the smart meter in real time," Torrico says.

While energy use is one common priority for homeowners, another is protecting property from hidden dangers such as undetected water leaks. Devlin Gualtieri wanted a water alarm system that could integrate several wireless units signaling a single receiver. But he didn't want to buy one designed to work with expensive home alarm systems charging monthly fees.

In this issue, Gualtieri writes about his wireless water alarm network, which has simple hardware including a Microchip Technology PIC12F675 microcontroller and water conductance sensors (i.e., interdigital electrodes) made out of copper wire wrapped around perforated board (p. 22).

It's an inexpensive and efficient approach that can be expanded. "Multiple interdigital sensors can be wired in parallel at a single alarm," Gualtieri says. A single alarm unit can monitor multiple water sources (e.g., a hot water tank, a clothes washer, and a home heating system boiler).

Also in this issue, columnist George Novacek begins a series on wireless data links (p. 58). His first article addresses the basic principles of radio communications that can be used in control systems.



Other issue highlights include advice on extending flash memory life (p. 54); using C language in FPGA design (p. 46); and detecting capacitor dielectric absorption (p. 62).

Mary Wilson

editor@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
Carlo van Nistelrooy

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
Ken Davidson, David Tweed, Chris Coulston

CUSTOMER SERVICE
Debbie Lavoie

US/UK
Carlo van Nistelrooy
+1 860 289 0800
c.vannistelrooy@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

circuit cellar — VOICE COIL — audioxpress — elektor

SUPPORTING COMPANIES

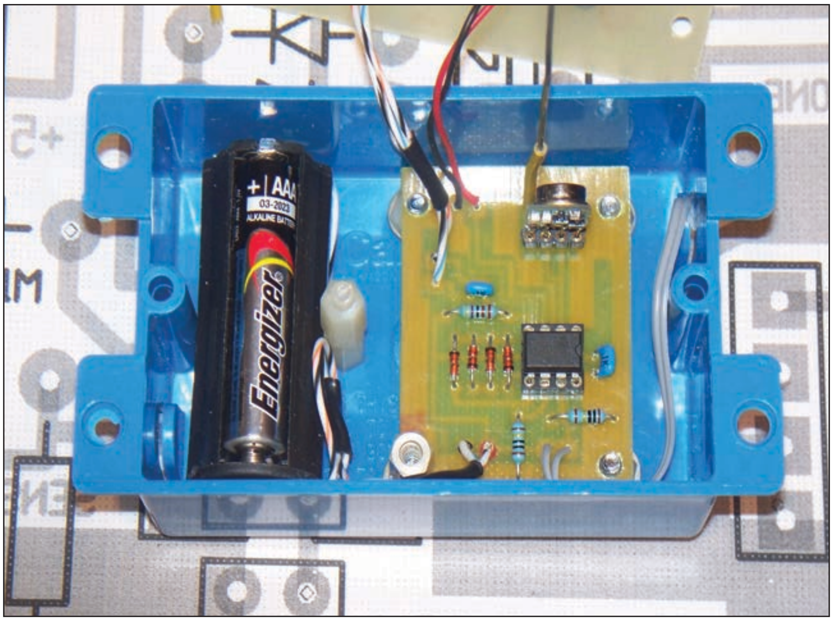
NOT A SUPPORTING COMPANY YET?

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

FEBRUARY 2014 • ISSUE 283

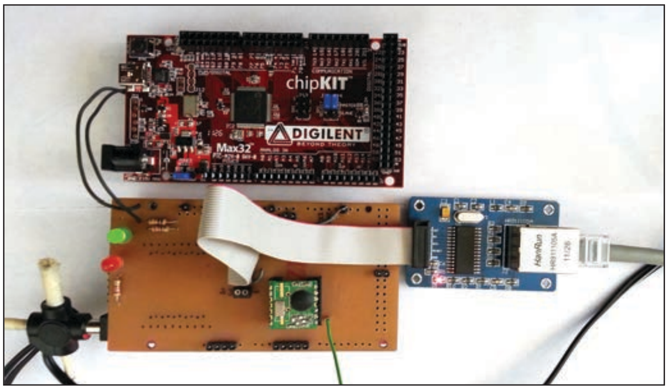
WIRELESS COMMUNICATIONS



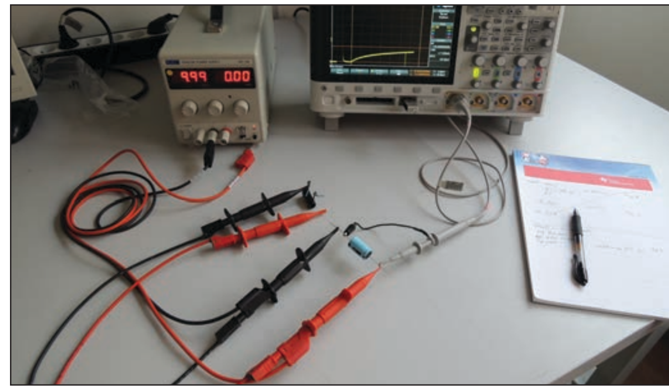
WIRELESS WATER ALARM WITH TRANSMITTER MODULE


SCOTT GARMAN AND HIS INTEL MINNOWBOARD ROBOTICS DEMO

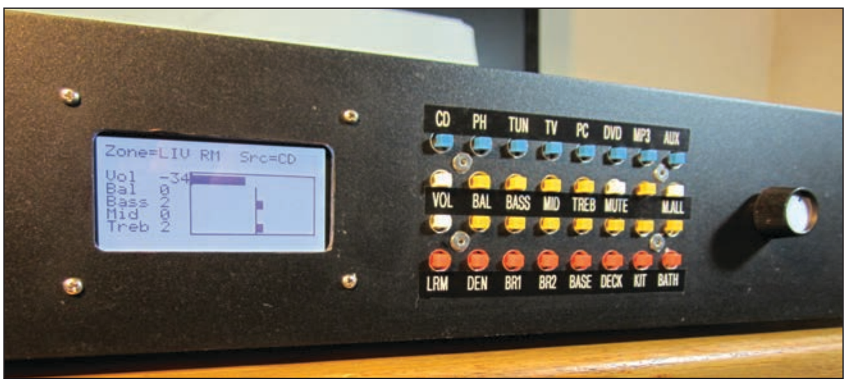
CONTENTS



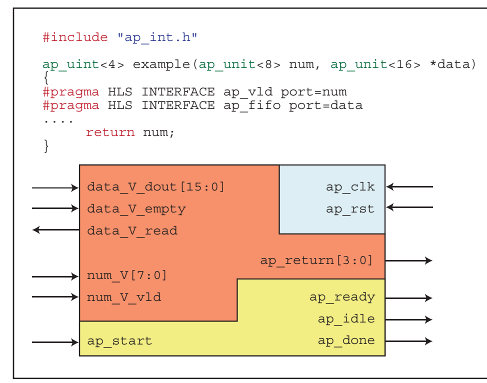
INNOVATIVE HOME ENERGY GATEWAY



DETECTING DIELECTRIC ABSORPTION



MULTI-ZONE AUDIO SYSTEM



MAPPING A C FUNCTION INTO AN FPGA BLOCK

COLUMNS

A POLE-CLIMBING ROBOT AT WORLD MAKER FAIRE (NEW YORK)

TESTS & CHALLENGES

TECH THE FUTURE

CC WORLD

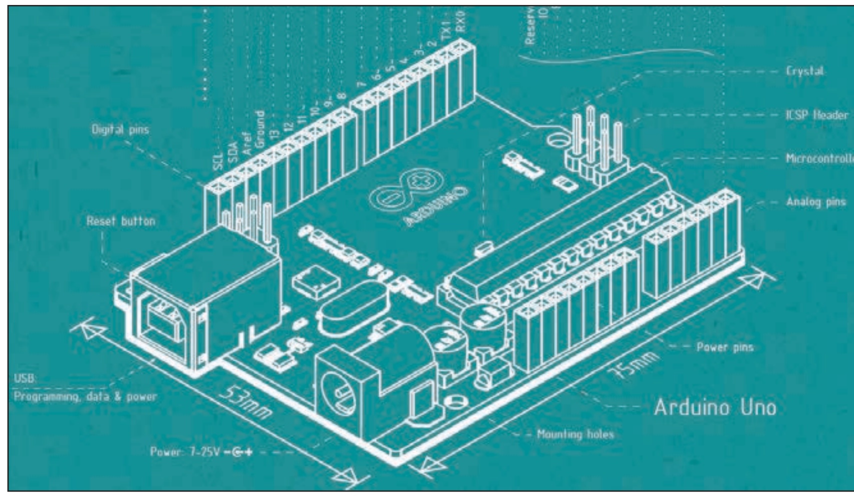
ARDUINO POSTER DOWNLOAD

by Elektor.Labs (US, UK, and The Netherlands)

Elektor.Labs recently produced an Arduino Uno blueprint poster. It details everything you need to know about the Arduino Uno. In addition to an interesting 3-D diagram, the poster includes detailed hardware specifications, information about "shields" and libraries, and notes about the Arduino IDE.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmel ATmega16U2 (ATmega8U2 up to version R2) programmed as a USB-to-serial converter. For Revision 3, the board has a stronger RESET circuit and an ATmega16U2 replaces the 8U2.

The Arduino Uno blueprint is Elektor. Labs's second poster. The first was a detailed, high-resolution photo of a Raspberry Pi.



Arduino is an open-source prototyping platform. The Arduino Uno's features include an Atmel ATmega, 5-V operating voltage, six PWM channels, six analog Input pins, and more.

- Download the Arduino Uno poster: http://bit.ly/1dxYnOR
- Download the Raspberry Pi poster: http://bit.ly/1d3s3FB

CC CODE CHALLENGE WINNER PAGE

by CC Staff (US)

```
module Ready_CrossClock(
    //First Clock Domain
    input wire CLK_A,
    input wire RDY_A,

    //Second Clock Domain
    input wire CLK_B,
    output wire RDY_B
);

/* Generate a signal which indicates RDY_A changing */
reg clkA_Change = 0;
always @(posedge CLK_A) begin
   clkA_Change <= clkA_Change ^ RDY_A;
end

/* Shift Register in B Clock Domain */
reg [2:0] ShiftReg_ClkB = 'b0;
always @(posedge CLK_B) begin
   ShiftReg_ClkB <= {clkA_Change, ShiftReg_ClkB[1:0]};
end

/* Output flag in B Clock domain */
assign RDY_B = (ShiftReg_ClkB[2] ^ ShiftReg_ClkB[1]);


endmodule
```

Engineers were challenged to spot the errors in code.

In 2013, *Circuit Cellar* managed the CC Code Challenge, which was sponsored by IAR Systems. The challenge ran for 31 weeks. Each week, *Circuit Cellar*'s technical editors purposely inserted an error into a snippet of code. Engineers were challenged to find the error for a chance to win prizes. All 31 challenges and winners are posted at http://bit.ly/18HO7pN.

The winners came from across the US and countries including the UK, Brazil, France, Indonesia, and The Netherlands. Winners were randomly selected in weeks when more than one entry was correct. Some entrants won twice during the year. The winners are listed here:

Peter Baston, Flintshire, UK; Gait Boxman, Gelderland, The Netherlands; Gordon Margulieux, Oregon, US; Alvin Schurman, Florida, US; Brian Shewan, Nova Scotia, Canada; Kang Usman, Jakarta, Indonesia; Guido Cargnino, Grugliasco (Province of Turin), Italy; John Safrit, Mebane, NC, US; Rob Tholl, Calgary, AB, Canada; Mike Brown, Meldreth, UK; Rudolf Steger, Braunschweig, Germany; Kevin Hannan, Marietta, GA, US; Laurent Haas, Paris, France; Carol Willing, Encinitas, CA, US; William McNamara, The Colony, TX, US; Jesper Poulsen, Copenhagen, Denmark; Carl Hage, California, US; Raffi Tchoboian, Vienne, France; Andrew Boelbaai, Texas, US; Michael Welle, Baden-Württemberg, Germany; Jon Chapman, Ohio, US; Marcelo Jimenez, Rio de Janeiro, Brazil; Alex Ivopol, Wellington, New Zealand; Colm Baston, Flintshire, UK; Antonios Chorevas, Attiki, Greece; Heinz Nickisch, Bavaria, Germany; Chu Tin Teng, Fremont, CA, US; and Chris Austen, South Yorkshire, UK.

QUESTIONS & ANSWERS

Technical Evangelist

An Interview with Scott Garman

Scott Garman is more than just a Linux software engineer. He is also heavily involved with the Yocto Project, an open-source collaboration that provides tools for the embedded Linux industry. In 2013, Scott helped Intel launch the MinnowBoard, the company's first open-hardware SBC.—Nan Price, Associate Editor

COMMUNITY

NAN: Describe your current position at Intel. What types of projects have you developed?

SCOTT: I've worked at Intel's Open Source Technology Center for just about four years. I began as an embedded Linux software engineer working on the Yocto Project and within the last year, I moved into a technical evangelism role representing Intel's involvement with the MinnowBoard.

Before working at Intel, my background was in developing audio products based on embedded Linux for both consumer and industrial markets. I also started my career as a Linux system administrator in academic computing for a particle physics group.

I'm definitely a generalist when it comes to working with Linux. I tend to bounce around between things that don't always get the attention they need, whether it is security, developer training, or community outreach.

More specifically, I've developed and maintained parallel computing clusters, created sound-level management systems used at concert stadiums, worked on multi-room home audio media servers and touchscreen control systems, dug into the dark areas of the Autotools and embedded Linux build systems, and developed fun conference demos involving robotics and computer vision. I feel very fortunate to be involved with embedded Linux at this point in history—these are very exciting times!

Scott was involved with an Intel MinnowBoard robotics and computer vision demo, which took place at LinuxCon Japan in May 2013.

NAN: Can you tell us a little more about your involvement with the Yocto Project (www.yoctoproject.org)?

SCOTT: The Yocto Project is an effort to reduce the amount of fragmentation in the embedded Linux industry. It is centered on the OpenEmbedded build system, which offers a tremendous amount of flexibility in how you can create embedded Linux distros. It gives you the ability to customize nearly every policy of your embedded Linux system, such as which compiler optimizations you want or which binary package format you need to use. Its killer feature is a layer-based architecture that makes it easy to reuse your code to develop embedded applications that can run on multiple hardware platforms by just swapping out the board support package (BSP) layer and issuing a rebuild command. New releases of the build system come out twice a year, in April and October.

I've maintained various user space recipes (i.e., software components) within OpenEmbedded (e.g., sudo, openssh, etc.). I've also made various improvements to our emulation environment, which enables you to run QEMU and test your Linux images without having to install it on hardware.

I created the first version of a security tracking system to monitor Common Vulnerabilities and Exposures (CVE) reports that are relevant to recipes we maintain. I also developed training materials for new

QUESTIONS & ANSWERS

developers getting started with the Yocto Project, including a very popular introductory screencast "Getting Started with the Yocto Project—New Developer Screencast Tutorial" (http://vimeo.com/36450321).

NAN: Intel recently introduced the MinnowBoard SBC. Describe the board's components and uses.

SCOTT: The MinnowBoard is based on Intel's Queens Bay platform, which pairs a Tunnel Creek Atom CPU (the E640 running at 1 GHz) with the Topcliff Platform controller hub. The board has 1 GB of RAM and includes PCI Express, which powers our SATA disk support and gigabit Ethernet. It's an SBC that's well suited for embedded applications that can use that extra CPU and especially I/O performance.

The MinnowBoard also has the embedded bus standards you'd expect, including GPIO, I^2C, SPI, and even CAN (used in automotive applications) support. We have an expansion connector on the board where we route these buses, as well as two lanes of PCI Express for custom high-speed I/O expansion.

There are countless things you can do with MinnowBoard, but I've found it is especially well suited for projects where you want to combine embedded hardware with computing applications that benefit from higher performance (e.g., robots that use computer vision, as a central hub for home automation projects, networked video streaming appliances, etc.).

And of course it's open hardware, which means the schematics, Gerber files, and other design files are available under a Creative Commons license. This makes it attractive for companies that want to customize the board for a commercial product; educational environments, where students can learn how boards like this are designed; or for those who want an open environment to interface their hardware projects.

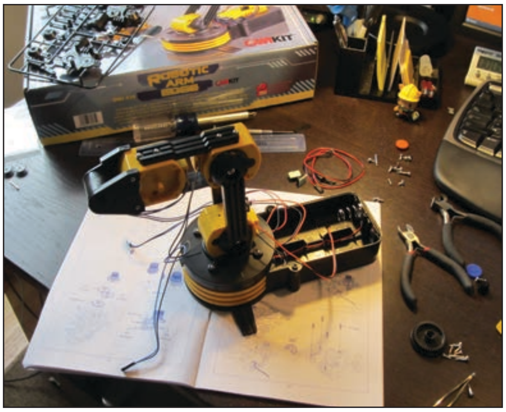
I created a MinnowBoard embedded Linux board demo involving an OWI Robotic Arm. You can watch a YouTube video to see how it works (www.youtube.com/watch?v=idSvJ-RXhq0).

NAN: What compelled Intel to make the MinnowBoard open hardware?

SCOTT: The main motivation for the MinnowBoard was to create an affordable Atom-based development platform for the Yocto Project. We also felt it was a great opportunity to try to release the board's design as open hardware. It was exciting to be part of this, because the MinnowBoard is the first Atom-based embedded board to be released as open hardware and reach the market in volume.

Open hardware enables our customers to take the design and build on it in ways we couldn't anticipate. It's a concept that is gaining traction within Intel, as can be seen with the announcement of Intel's open-hardware Galileo project (www.intel.com/content/www/us/en/do-it-yourself/galileo-maker-quark-board.html).

Scott is shown working on an Intel MinnowBoard demo, which was built around an OWI Robotic Arm.

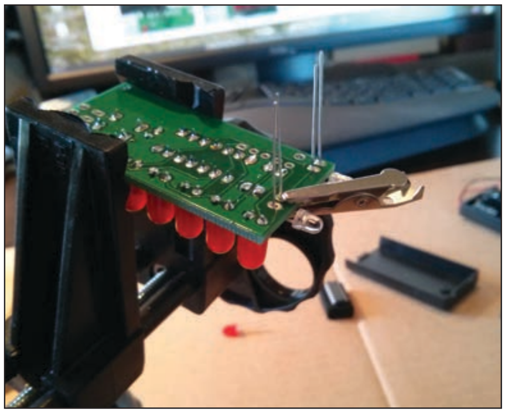


Here, the OWI Robotic Arm is being assembled.

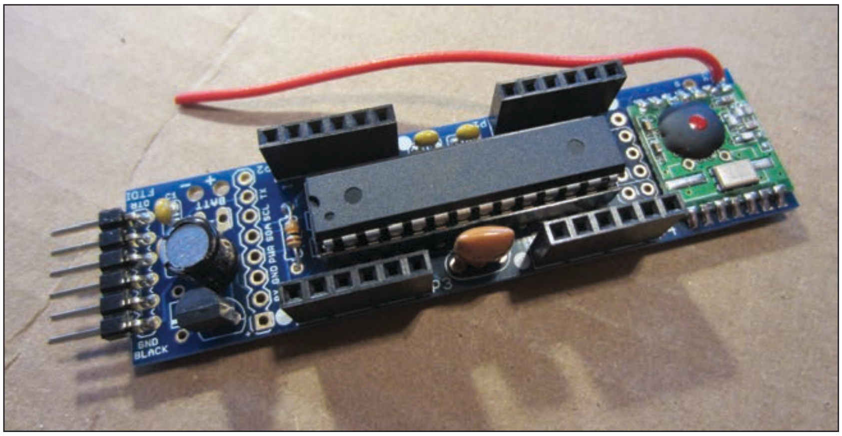
Scott doesn't have a dedicated workbench or garage. He says he tends to just clear off his desk, lay down some cardboard, and work on things such as the Trippy RGB Waves Kit, which is shown.

QUESTIONS & ANSWERS

Scott's Blinky POV Kit is shown. "I don't know what I'd do without my PanaVise Jr. [vise] and some alligator clips," he said.



Below is a completed JeeNode v6 Kit Scott built one weekend.



NAN: What types of personal projects are you working on?

SCOTT: I've recently gone on an electronics kit-building binge. Just getting some practice again with my soldering iron with a well-paced project is a meditative and restorative activity for me.

I worked on one project, the Trippy RGB Waves Kit, which includes an RGB LED and is controlled by a microcontroller. It also has an IR sensor that is intended to detect when you wave your hand over it. This can be used to trigger some behavior of the RGB LED (e.g., cycling the colors). Another project, the Blinky POV Kit, is a row of LEDs that can be programmed to create simple text or logos when you wave the device around, using image persistence.

My current project is to add some wireless sensors around my home, including temperature sensors and a homebrew security system to monitor when doors get opened using 915-MHz JeeNodes. The JeeNode is a microcontroller paired with a low-power RF transceiver, which is useful for home-automation projects and sensor networks. Of course the central server for collating and reporting sensor data will be a MinnowBoard.

NAN: Tell us about your involvement in the Portland, OR, open-source developer community.

SCOTT: Portland has an amazing community of open-source developers. There is an especially strong community of web application developers, but more people are hacking on hardware nowadays, too. It's a very social community and we have multiple nights per week where you can show up at a bar and hack on things with people.

I'd say it's a novelty if I wasn't so used to it already—walking into a bar or coffee shop and joining a cluster of friendly people, all with their laptops open. We have coworking spaces, such as Collective Agency, and hackerspaces, such as BrainSilo and Flux (a hackerspace focused on creating a welcoming space for women).

Take a look at Calagator (calagator.org) to catch a glimpse of all the open-source and entrepreneurial activity going on in Portland. There are often multiple events going on every night of the week. Calagator itself is a Ruby on Rails application that was frequently developed at the bar gatherings I referred to earlier. We also have technical conferences ranging from the professional OSCON to the more grassroots and intimate Open Source Bridge (opensourcebridge.org).

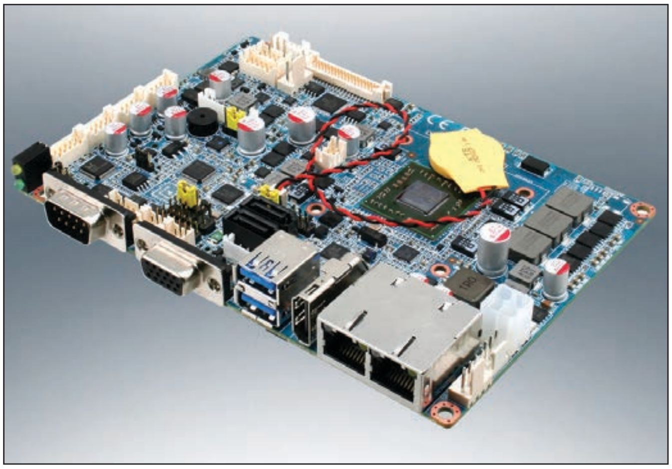
I would unequivocally state that moving to Portland was one of the best things I did for developing a career working with open-source technologies, and in my case, on open-source projects.

This photo was taken in the Open Source Bridge hacker lounge, where people socialize and collaborate on projects. Here someone brought a brainwave-control game. The players are wearing electroencephalography (EEG) readers, which are strapped to their heads. The goal of the game is to use biofeedback to move the floating ball to your opponent's side of the board.

PRODUCT NEWS

AMD EMBEDDED G-SERIES SoC SOLUTION

The **ECM-KA** SBC is powered by the AMD Embedded G-Series first-generation system-on-chip (SoC) accelerated processing unit (APU), which is based on 28-nm design technology. The AMD processors are built on Jaguar microarchitecture and integrate a quad-core CPU and a next-generation graphics core.



The small-footprint ECM-KA provides extremely low power consumption, high graphic performance, multimedia, and I/O. The SBC is designed for embedded applications including industrial controls and automation, gaming, thin clients, retail/digital signage, SMB storage server, surveillance, medical, communication, entertainment, and data acquisition.

The ECM-KA supports one 204-pin DDR3 SODIMM socket that provides up to 8 GB DDR3 1600 SDRAM. It also accommodates dual-channel 18-/24-bit low-voltage differential signaling (LVDS) as well as HDMI and VGA multi-display configurations. The I/O deployment includes two SATA III, one mini PCIe, one CF, two USB 3.0, six USB 2.0, two COM, 8-bit DIO, and 2-Gb Ethernet. Multiple OS support—including Windows 8, Windows 7, and Linux—can be used in various embedded designs.

Contact Avalue for pricing.

Avalue Technology, Inc.
www.avalue.com.tw

HAND-HELD DSOs FOR INDUSTRIAL APPLICATIONS

The **2510** series of hand-held digital storage oscilloscopes (DSOs) includes 60- and 100-MHz bandwidth models in non-isolated and isolated designs. The portable, dual-channel DSOs provide floating measurement capabilities with built-in digital multimeter (DMM) and recorder functions.

Models 2511 and 2512 provide non-isolated 300-V CAT II-rated inputs and are well suited for general electronics. Designed for industrial applications, models 2515 and 2516 provide two fully isolated 1,000-V CAT II/600-V CAT III-rated inputs and feature fully isolated channels for safe and accurate floating measurements.

All 2510 series models offer 5.7″ color displays, a built-in 6,000-count DMM with dedicated terminals for current measurement, and oscilloscope/meter trend plot functions to log measurement values. You can use the DSOs to capture and record up to two voltage or time parameters or graph any of the DMM's measurements, including DC/AC voltage and current, resistance, capacitance, diode, and continuity testing.

The 2510 DSOs' additional features include digital filtering with adjustable limits, 32 automatic measurements, FFT and math functions, a multi-language interface, and USB connectivity for remote PC control. A side-panel USB host port is available to save setups, waveform data, screenshots, or CSV files to a USB flash drive.

Each model weighs approximately 3.4 lb with the included rechargeable lithium-ion battery and can provide up to 4 h of continuous battery operation. The DSOs include two high-bandwidth, safety-rated passive probes and a pair of DMM test leads. Models 2515 and 2516 include a hard plastic carry case accessory.

The 2510 series models are backed by a standard three-year warranty. The model 2511 (60 MHz, two non-isolated channel, 300-V CAT II) costs **$935**. The model 2512 (100 MHz, two non-isolated channel, 300-V CAT II) costs **$1,125**. The model 2515 (60 MHz, two isolated channel, 1,000-V CAT II, 600-V CAT III) costs **$2,165**. The model 2516 (100 MHz, two isolated channel, 1,000-V CAT II, 600-V CAT III) costs **$2,395**.

B&K Precision Corp.
www.bkprecision.com

PRODUCT NEWS

ULTRA-COMPACT CONVERTER

The **PQA50-D** is an ultra-compact 50-W DC/DC converter that reduces board space and costs in telecommunications, industrial, and IT equipment applications. The DC/DC converter family features a 2" × 1" (50.8-mm × 25.4-mm) footprint and incorporates six-sided metal shielding for improved electromagnetic compatibility (EMC) performance and up to 93% efficiency.

The single-output isolated DC/DC converter modules feature a 2:1 input range and are available with an 18~36 VDC or 36~75 VDC input voltage range and 3.3-, 5-, 12-, 15-, or 24-VDC output. The series features 1,500-VDC I/O isolation and protections for output over voltage, short circuit, overload, and input under voltage.

The PQA50-D converters offer precise voltage regulation, featuring load regulation of ±1% maximum from 10% to 100% load and line regulation of ±0.5% maximum. The converters also feature a −40~85°C operating temperature range, remote on/off control, and ±10% voltage adjustability.

Pricing for the PQA50-D converters starts at **$85.12** in 100-unit quantities.

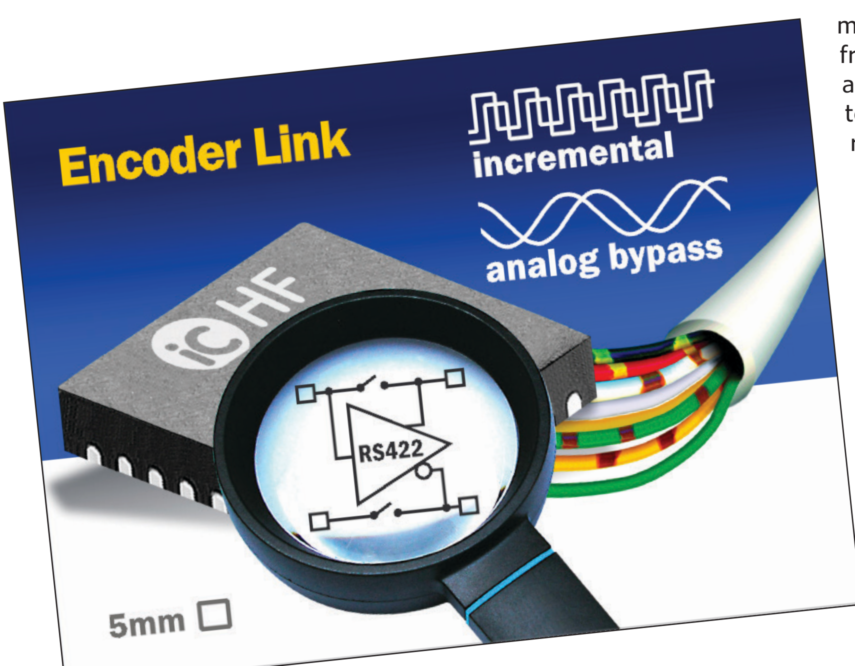
CUI, Inc.
www.cui.com

SIX-CHANNEL RS-422 LINE DRIVER/RECEIVER

The **iC-HF** provides six RS-422 line drivers for 3-to-5.5-V encoder applications. The device contains reverse polarity protection for a safe sensor-side supply of up to 60 mA.

The iC-HF is pin configurable. A safe external signal sequence at two complementary line driver outputs activates the Encoder Link state, in which nine pins are connected from the sensor side to the field side. With this low-impedance bypassing, internal analog sensor signals and digital communication signals (e.g., BiSS, SPI, I²C, etc.) can be accessed at the line driver output pins.

With the integrated Encoder Link function, the line drivers can be deactivated and A/D signals can be directly accessed through the line driver output pins. Conventional sensors can be calibrated or programmed through the usual RS-422 outputs via the Encoder Link function. Extra contacts, pins, control lines, or signals are not needed. For differential RS-422 line driver operation, six differential complementary drivers are implemented.



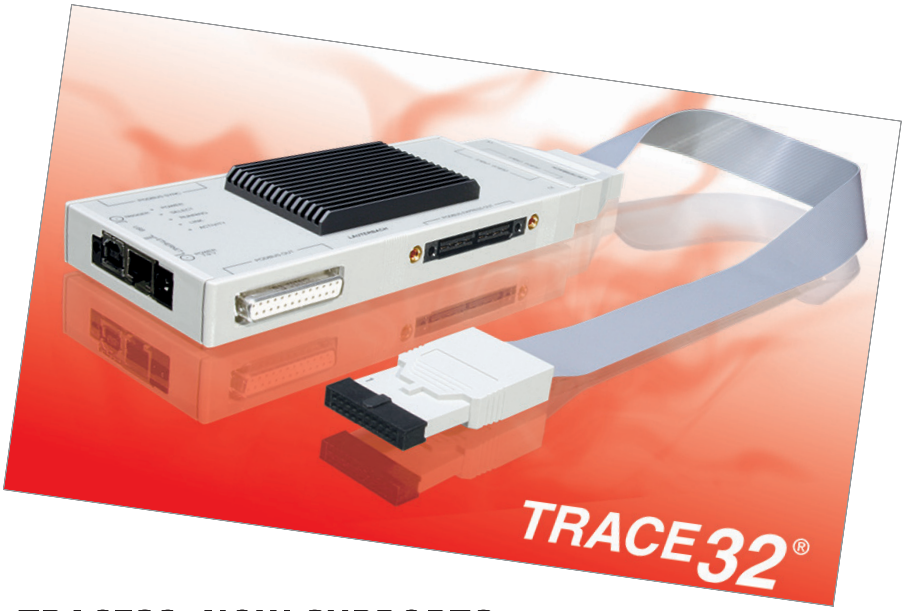
Each push-pull driver stage can drive up to 65 mA maximum at 5 V and operate at up to a 10-MHz output frequency with RS-422 termination. The driver stages are current limited, short-circuit proof, and over-temperature protected. The current limitation also reduces electromagnetic compatibility (EMC).

The device provides undervoltage detection and on-chip temperature monitoring to switch the driver stages to high impedance on demand. A sensor error signal is combined with the iC-HF error states. When a fault occurs, the open-drain error output NERR is activated. All inputs are CMOS- and TTL-compatible and ESD protected.

The iC-HF's operating temperatures range from −40°C to 125°C. The device is available in a 5-mm × 5-mm 32-pin QFN package. The design-in process is supported by ready-to-operate demonstration boards including the Encoder Link signal sequence generator.

The iC-HF costs **$3.65** in 1,000-unit quantities.

iC-Haus GmbH
www.ichaus.com



TRACE32 NOW SUPPORTS XILINX MICROBLAZE 8.50.C

TRACE32 modular hardware and software supports up to 350 different CPUs. The microprocessor development tools now complement the latest version of Xilinx's MicroBlaze 8.50.c, which is a soft processor core designed for Xilinx FPGAs. The MicroBlaze core is included with Xilinx's Vivado Design Edition and IDS Embedded Edition.

The TRACE32 tools have supported MicroBlaze for many years by providing efficient and user-friendly debugging at the C or C++ level using the on-chip JTAG interface. This interface also provides code download, flash programming, and quick access to all internal chip peripherals and registers.

Contact Lauterbach for pricing.

Lauterbach GmbH
www.lauterbach.com

Xilinx, Inc.
www.xilinx.com

HIGH-ACCURACY SNOW DEPTH SENSOR

MaxBotix **Snow Depth Sensors** operate in −40C° to 65C° temperatures with a 5,000-mm maximum range. The sensors include responsive accurate temperature compensation. They perform well during heavy snowfall conditions and provide continued performance during high-wind conditions.

The sensors feature a 200,000-h mean time between failure (MTBF). Their low-power consumption enables them to operate in battery-based systems for extended periods of time (or solar-powered systems indefinitely), which eases maintenance requirements and enables remote installations.

Six different Snow Depth Sensors are available. The sensors are equipped for the following interface controls: pulse width, analog voltage, and serial data RS-232 (MB7354) or TTL (MB7374).

The Snow Depth Sensors cost **$149.99**.

MaxBotix, Inc.
www.maxbotix.com

PRODUCT NEWS

COMPACT DC/DC CONVERTERS

The **JCE06** and **JTE06** series of low-cost compact 6-W DC/DC converters are designed for a range of mobile communications and industrial and transportation applications. The converters are constructed from a plastic industry-standard 24-pin DIP through-hole mounting package. They measure 1.25″ × 0.80″ × 0.4″ (31.75 mm × 20.32 mm × 10.40 mm). The single- and dual-output units achieve a power density of 15 W per in^3 and up to 84% efficiency.

Single-output models are available with 3.3-, 5-, 12-, 15-, or 24-VDC outputs. Dual-output models provide ±3.3-, ±5-, ±12-, ±15-, or ±24-VDC outputs. The JCE06 accommodates a wide 2:1 input range. The JTE06 series is designed for an ultra-wide 4:1 input range. All models cover the popular nominal input 12-, 24-, or 48-VDC voltages.

I/O isolation is 1,500 VDC across the range. The converters suit a range of operating environments in temperatures from −40°C to 100°C. No additional forced airflow or heatsinking is required, which saves valuable board space and additional cost.

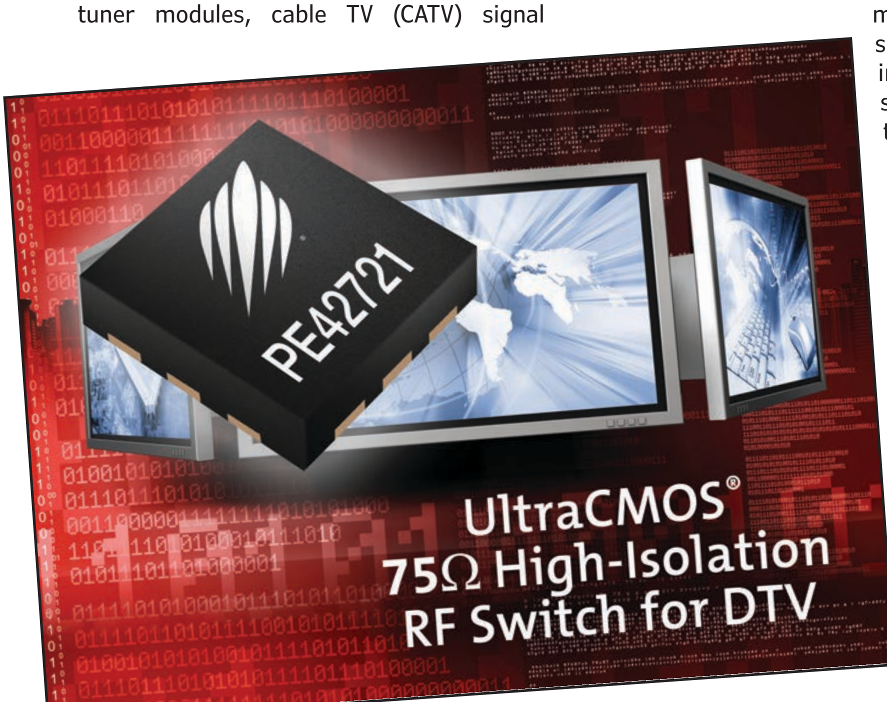
An under-voltage input protection feature shuts down the converter below a set voltage to prevent damage to the converter. All models within both ranges meet EN55022 Class-A conducted electromagnetic compatibility (EMC) emissions without any additional components.

Pricing for the single-output JCE06 series starts at **$13.85** in 500-unit quantities. Pricing for the JTE06 series starts at **$14** in 500-unit quantities.

XP Power, Ltd.
www.xppower.com

RF SWITCH FOR TVs, SET-TOP BOXES, AND DVRs

The **PE42721** is an absorptive 75-Ω single-pole, double-throw (SPDT) RF switch developed on Peregrine's UltraCMOSprocess technology. The Peregrine HaRP technology-enhanced switch is well suited for designers of digital TV (DTV) tuner modules, cable TV (CATV) signal switching and distribution systems, multi-tuner DVRs, and set-top boxes.



The switch is ideal for geographic markets (e.g., China, Japan, Korea, and Latin America) where broadband-TV devices must accommodate multiple RF inputs for cable, satellite, and terrestrial reception while avoiding interference among these signals. The PE42721 switch's isolation is very high; therefore, only one tuner chip is needed in each device.

The switch pairs the high isolation with low insertion loss, which improves the signal fidelity by reducing the signal-to-noise ratio (SNR). This feature also helps the switch deliver best-in-class linearity and electrostatic-discharge (ESD) performance.

The PE42721 provides isolation of greater than 55 dB across the entire broad-frequency range from 5 MHz to 2.2 GHz. The 3-mm × 3-mm switch supports 1.8- and 3.3-V control logic and RoHS-compliant packaging.

The PE42721 costs **$0.45** each in 10,000-unit quantities. Evaluation kit boards cost **$95**.

Peregrine Semiconductor Corp.
www.psemi.com

METCAL LAUNCHES NEW MX-500

The latest version of Metcal's **MX-500** soldering and rework system features ground fault interrupt and universal power supply in a new housing. The system is compatible with existing MX-500 products, including MX upgrade kits, tip cartridges, and accessories.

Built on Metcal's SmartHeat technology, the new MX-500 retains switchable dual-port 40-W operation. The system provides responsive and highly controlled heating and delivers the exact energy needed to ensure a precise and reliable solder connection.

Contact Metcal for pricing.

Metcal
www.metcal.com

ARM CORTEX A8 SYSTEM ON MODULE

The **M-5360A** is an application-ready solution for multimedia and machine-to-machine (M2M) applications. The credit card-size System on Module (SOM) is powered by a Freescale Semiconductor 800-MHz i.MX537 ARM Cortex A8 processor with 1-GB DDR3 RAM and 4-GB eMMC flash.

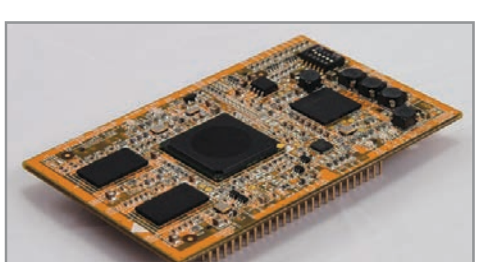
The M-5360A features two independent low-voltage differential signaling (LVDS) channels for dual LCDs and one VGA port for external monitor connection. The i.MX537's multimedia and graphic engine supports OpenGLE 2.0, OpenVG 1.1 graphics acceleration, and 1080P video decoding.

The M-5360A also provides powerful communication functionality (e.g., Ethernet, RS-232, RS-485, CAN 2.0, 1-Wire, and USB). This makes the SOM suitable for multimedia applications as well as embedded networking devices.

The M-5360A uses 128-pin 2-mm pin headers, which simplifies application board design. The SOM includes a preinstalled Ubuntu OS. Android and Windows CE are available by request. In addition to the hardware building blocks, software utility and device drivers are available for user applications.

Contact Artila for pricing.

Artila Electronics Co., Ltd.
www.artila.com



CLIENT PROFILE

Invenscience LC

www.invenscience.com

2340 South Heritage Drive, Suite I, Nibley UT, 84321

CONTACT: Collin Lewis, sales@invenscience.com

EMBEDDED PRODUCTS: Torxis Servos and various servo controllers

FEATURED PRODUCT: Invenscience features a wide range of unique servo controllers that generate the PWM signal for general RC servomotors of all brands and Torxis Servos. (The Simple Slider Servo Controller is pictured.) Included in this lineup are:

- Gamer joystick controllers
- Conventional joystick controllers
- Equalizer-style slider controllers
- Android device Bluetooth controllers

All of these controllers provide power and the radio control (RC) PWM signal necessary to make servos move without any programming effort.

EXCLUSIVE OFFER: Use the promo code "CC2014" to receive a 10% discount on all purchases through March 31, 2014.

Circuit Cellar prides itself on presenting readers with information about innovative companies, organizations, products, and services relating to embedded technologies. This space is where Circuit Cellar enables clients to present readers useful information, special deals, and more.

FEATURES

Build an Inexpensive Wireless Water Alarm

Want to prevent water damage in your home or building? Consider this inexpensive, battery-operated water alarm. It features a wireless receiver, audible alarm, and low-battery alert.

By Devlin Gualtieri (US)

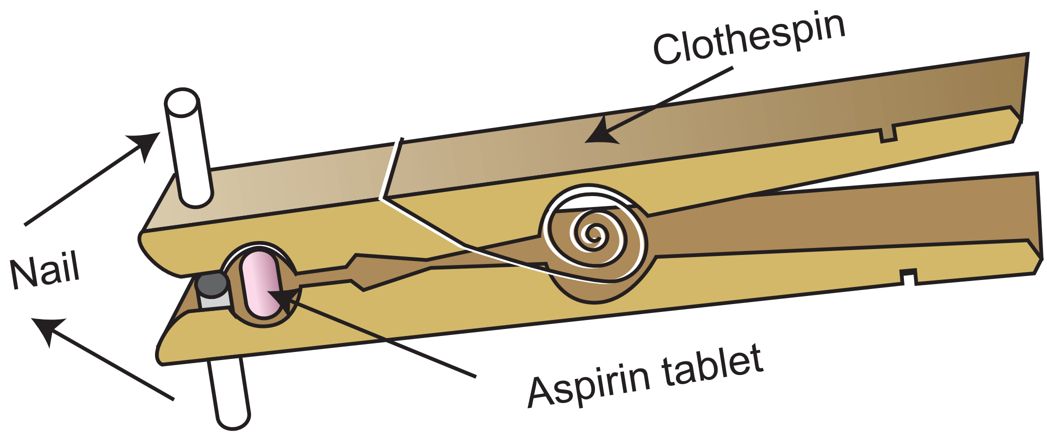


FIGURE 1
This is a simple water sensor, suitable for getting your elementary school child interested in electricity. When water dissolves the aspirin tablet, the clothespin snaps shut to complete a circuit wired to the nail electrodes.

Water problems are a foremost homeowner complaint. Aside from Hurricane Sandy, which caused major problems in my northern New Jersey locale, I've had problems with a leaking storage tank for my well water, a leaking hot water heater, a cracked boiler on my home heating system, an overflowing dehumidifier, and water seeping under the cellar bulkhead doors after some landscaping caused water to flow toward rather than away from a section of the house.

I'm sure other homeowners have had similar water problems. I've never had an overflowing sump, but many homeowners can add that to their water problem list.

In all cases, an early warning about water on the floor would have prevented a lot of the resulting damage. That got me thinking about water alarms. It's possible to purchase some simple and inexpensive battery-powered water alarms. A quick online search will reveal quite a few. From customer reviews, they all seem to work quite well. However, after scribbling some specifications for the type of alarm I really wanted and how many I needed, I quickly decided to design my own.

Primarily, I wanted a bunch of wireless units that signaled a single receiver. You can purchase these types of units, but they're designed to work with proprietary and expensive home alarm systems, many of which make extra money from a monthly "monitoring" fee. As an advocate of free and open-source software (FOSS), it's not surprising that I object to such schemes. Owning a soldering iron gives electronic hobbyists some choices other people don't have.

WATER SENSING

When I was a young boy in the 1950s, quite a few magazines featured simple electrical projects a child could build to occupy his time. One of these was a clothespin water alarm (see **Figure 1**).

The design worked by using water to dissolve an aspirin tablet holding a clothespin's jaws apart. When the clothespin snapped shut, two nails would contact to complete a circuit with a battery and an alarm bell. This circuit had a battery life that was essentially the battery's shelf life. The downside was that it needed to be reset after each use by replacing the aspirin tablet.

This simple idea was also used in a door alarm I received as a gift several years ago. In that case, a cord pulled an insulating tab from a similar mechanism.

One electrical way to detect water is by its dielectric constant, which is huge. Glass has a dielectric constant of about 4. Tantalum oxide, the magic component of tantalum capacitors, has the exceptionally high dielectric constant of 25. Water, because of the polarizability of the water molecules, has a dielectric constant of about 80. For this reason, it's easy to detect water when it displaces air in a capacitance cell.

Another detection method is to monitor water's conductance. Even pure water conducts electricity since it has a small concentration of hydrogen (actually hydronium) ions and hydroxide ions in equilibrium with the water molecules. Impure water has better conductivity because of dissolved electrolytes from minerals. Circuits designed to detect water by its conductivity are usually simpler than those designed for the capacitance method.

The simplest geometry for a water conductance sensor is the interdigital electrode. This is especially convenient since the sensor sits flat on a surface to detect the first water. This type of geometry lends itself to a PCB pattern, but there's an easier way to make such an electrode: You can simply wind two parallel coils of 22 AWG copper wire on a perforated board about 2" × 4" (see **Photo 1**).

Although it is easiest using bare copper wire, I used enameled copper wire, simply because that's what I had. In that case, I had to remove the enamel from the flat area on one side of the perforated board. I used some sand paper and did a little scraping with an X-ACTO knife to accomplish this task.

Exposure to tap water, which is well water in my house, causes a resistance change of the sensor from presumed infinity (more likely several hundred megohms) to about 50 kΩ. This is more than enough change for a useful voltage signal. In the alarm circuit, this change results in a voltage change at the microcontroller's high-impedance analog input pin from 0 V to nearly the entire circuit supply voltage.

LOW POWER

Low power occurs when a microprocessor is in its Sleep mode, but you still need to do a few things to get the sleep power as low as possible. One trick to low-power operation is not to have input pins float between the power supply rails. You can ground unused input pins, but it's easiest to just set unused pins as output pins set at either rail without any connection. For a Microchip Technology PIC microcontroller, it's essential to disable the brown-out voltage detection circuitry. This can add nearly 100 µA to the current drain! Also, disable unneeded functions (e.g., voltage comparators, etc.).

When the water alarm is in the Sleep mode, the current draw is less than 10 µA. A quick calculation using 1,000 mAh for an alkaline cell capacity gives you 100,000 h (more than 10 years). This may be a little optimistic, but it is encouraging. Of course, when the ADCs are operational, there's a larger power draw.

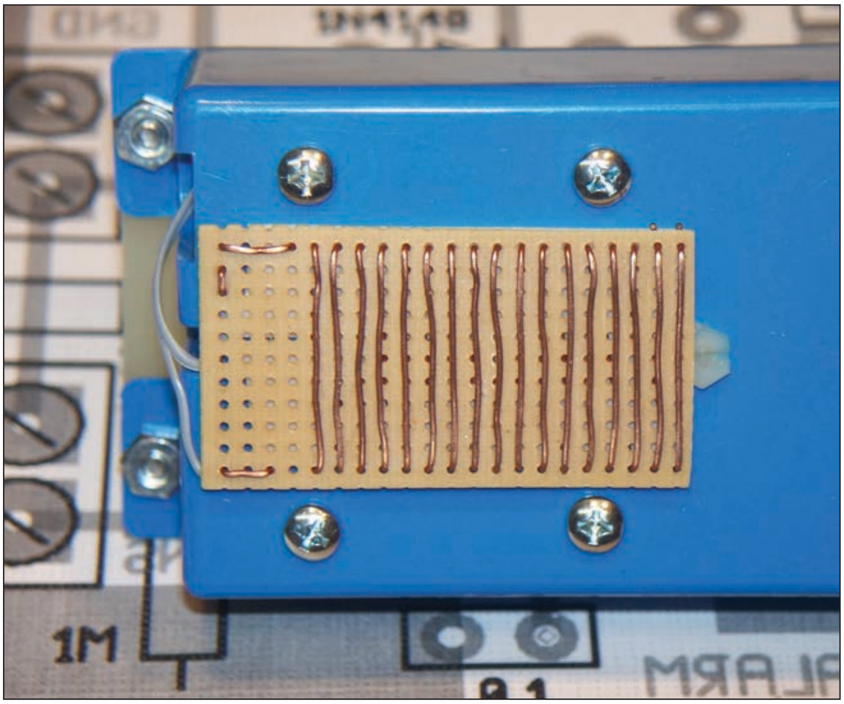


PHOTO 1
An interdigital water detection sensor is shown. Alternate rows are lengths of AWG 22 copper wire, which is either bare or has its insulation removed. The sensor is shown mounted to the bottom of the box containing the water alarm circuitry. I attached it with double-stick foam tape, but silicone adhesive should also work.

Since this rarely happens (I set the update rate for water detection at 15 min and for battery monitoring at 24 h), the batteries should last their shelf lives (or not, since I haven't done a 10-year test).

BATTERY MONITOR

A battery monitor is an important feature of any battery-powered alarm circuit. The Microchip Technology PIC12F675 microcontroller I used in my alarm circuit has 10-bit ADCs that can be optionally assigned to the I/O pins. However, the problem is that the reference voltage for this conversion comes from the battery itself. As the battery drains from 100% downward, so does the voltage reference, so no voltage change would be registered.

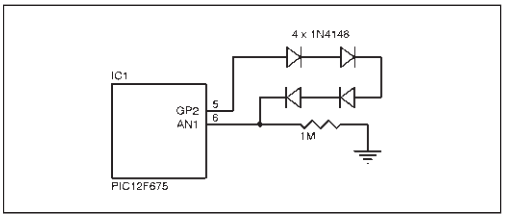


FIGURE 2
This is the portion of the water alarm circuit used for the battery monitor. The series diodes offer a 1.33-V total drop, which offers a reference voltage so the ADC can see changes in the battery voltage.

Battery Voltage	ADC Value
5	751
4.75	737
4.5	721
4.24	704
4	683
3.75	661

TABLE 1
The battery voltage and ADC values are shown.

Center Frequency	Width of Band	Notes
6.78 MHz	30 kHz	–
13.56 MHz	14 kHz	–
27.12 MHz	326 kHz	–
40.68 MHz	40 kHz	–
433.92 MHz	1.84 MHz	Popular alarm frequency
915 MHz	26 MHz	US Only
2.45 GHz	100 MHz	WLAN and microwave ovens
5.8 GHz	150 MHz	WLAN

TABLE 2
Information about lower-frequency industrial, scientific, and medical (ISM) bands is shown.

I used a simple mathematical trick to enable battery monitoring. **Figure 2** shows a portion of the schematic diagram. As you can see, the analog input pin connects to an output pin, which is at the battery voltage when it's high through a series connection of four small signal diodes (1N4148). The 1-MΩ resistor in series with the diodes limits their current to a few microamps when the output pin is energized. At such low current, the voltage drop across each diode is about 0.35 V. An actual measurement showed the total voltage drop across the four diodes to be 1.33 V.

This voltage actually presents a new reference value for my analog conversion. The analog conversion now provides the following digital values:

$$V_{ADC} = 1{,}024\left[1 - \left(\frac{1.33}{V_{BAT}}\right)\right]$$

Table 1 shows the digital values as a function of battery voltage. The nominal voltage of three alkaline cells is 4.75 V. The nominal voltage of three lithium cells is 5.4 V. The PIC12F675 functions from approximately 2 to 6.5 V, but the wireless transmitter needs as much voltage as possible to generate a reliable signal. I arbitrarily coded the battery alarm at 685, or a little above 4 V. That way there's still enough power to energize the wireless transmitter at a useful power level.

WIRELESS TRANSMITTER

The International Telecommunications Union (ITU) has set aside frequencies up and down the radio bands for "industrial, scientific, and medical" (ISM) applications. Wireless LAN frequencies are among these, as is the frequency of microwave ovens, primarily so they won't interfere with more important signals.

Table 2 lists the lower-frequency ISM bands. Other frequencies are set aside for similar applications in the US (e.g., 315 MHz is used for garage door transmitters).

A Wi-Fi wireless link would be one signaling solution. The problem with this is that you would need a dedicated computing device on the receiver end. This could be done with an inexpensive SBC (e.g., the Raspberry Pi). Of course, you would need to buy a USB-compatible Wi-Fi device and write some code.

Since my application is just signaling an alarm, a simple transmitter and a receiver that detects the presence of the signal is all that is needed. There are inexpensive modules available for both transmission and reception on the ISM bands. A quarter-wave antenna at the 916.5-MHz frequency I used is just 8 cm. Its relatively low frequency enables penetration of radio waves through most residential walls and flooring. A quarter wavelength at 433 MHz is 17 cm.

Photo 2 shows one of the transmitter modules I used in my system. The round device is a surface acoustic wave (SAW) resonator. It just takes a few components to transform this into a low-power transmitter operable over a wide supply voltage range, up to 12 V. The companion receiver module is also shown. My alarm has a 916.5-MHz operating frequency, but 433 MHz is a more popular alarm frequency with many similar modules.

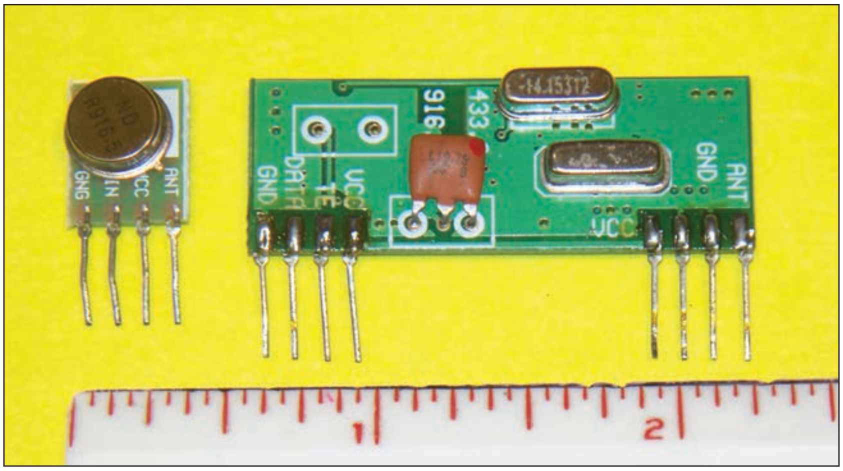


PHOTO 2
These transmitter and receiver modules are used in the water alarm. The modules operate at 916.5 MHz, but 433 MHz is a more common alarm frequency with similar modules. The scale is inches.

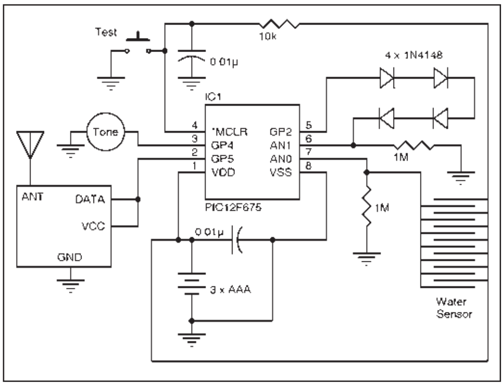


FIGURE 3
The water alarm circuit includes instructions for fabricating the interdigital electrode.

FEATURES

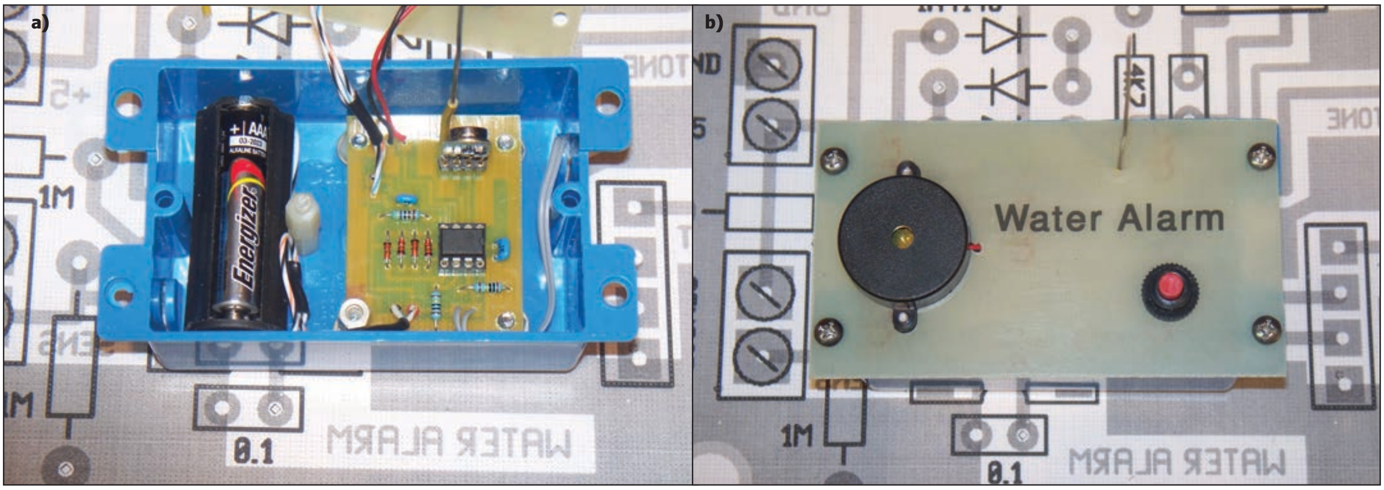


PHOTO 3
a—This is the water alarm's interior. The transmitter module with its antenna can be seen in the upper right. The battery holder was harvested from a $1 LED flashlight. The box is 2.25″ × 3.5″, excluding the tabs. **b—**The water alarm is shown with the top panel in place. The antenna pokes out of the panel, which was made from a piece of copper-clad board with the copper removed. The acoustic resonator is seen on the left. The reset button is shown at the lower right.

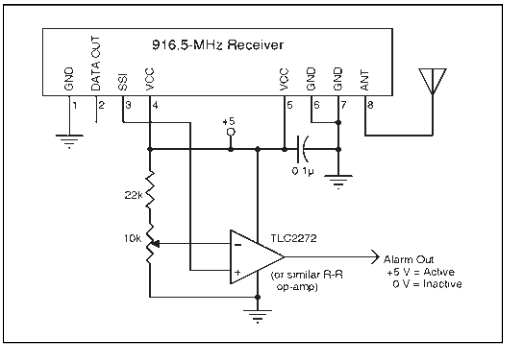


FIGURE 4
The water alarm wireless receiver is shown. The potentiometer sets the threshold for the digital output signal, which is derived from the analog signal strength indicator (SSI).

ALARM CIRCUIT

Figure 3 shows the alarm circuit. I've already discussed the battery monitor operation and water sensor portion. I used three alkaline AAA batteries, simply because I could harvest battery holders from some inexpensive LED flashlights. Alternatively, AA batteries can be used. There's really no advantage to using lithium batteries in this application.

The Test button simply resets the microcontroller to start the stored program again. There's code at the start to sound a short alarm and send a wireless signal. The latter is good to verify that you are getting reception at your receiver location. The tone element is just a piezoelectric element in a Helmholtz acoustic resonator. These are sold as inexpensive units in various sizes from various sources. The Helmholtz acoustic cavity amplifies the audio signal.

Helmholtz resonators work best when excited at their resonant frequency, which is usually 2.5 kHz. The software generates this excitation frequency, and the 5-V square-wave drive provides ample volume. Note that the alarm signal is not quite as loud as that of a smoke alarm. It's just there to identify which unit detects water when the receiver gets a signal. With a small source code modification to output an unmodulated signal, you can add a louder, buzzer-type alarm if you wish.

The voltage supply for the transmitter is from a microcontroller output pin, so the transmitter doesn't have any power drain when not needed. These transmitters usually have a data pin, which is held high to turn on the transmitter. **Photo 3** shows how the alarm circuitry and battery holder fit into a small electrical utility box.

RECEIVER CIRCUIT

Receiver modules are designed for data reception, so in the water alarm application I needed to add some additional circuitry

PROJECT FILES

circuitcellar.com/ccmaterials

RESOURCES

International Telecommunication Union, www.itu.int.

Microchip Technology, Inc., "PIC12F629/675 Datasheet," 2010.

———, "Compiled Tips 'N Tricks Guide, Chapter 2: PIC Microcontroller Low Power Tips 'n Tricks," 2009.

Wikipedia, "ISM band."

SOURCE

PIC12F675 Microcontroller
Microchip Technology, Inc. | www.microchip.com

ABOUT THE AUTHOR

Devlin Gualtieri, who lives in Legdewood, NJ, received his PhD in Solid-State Science and Technology from Syracuse University in 1974. He had a 30-year career in research and technology at a major aerospace company and is now retired. Devlin writes a science and technology blog (www.tikalon.com/blog/blog.php) and is the author of two science fiction novels. For more information, visit www.tikalonpress.com.

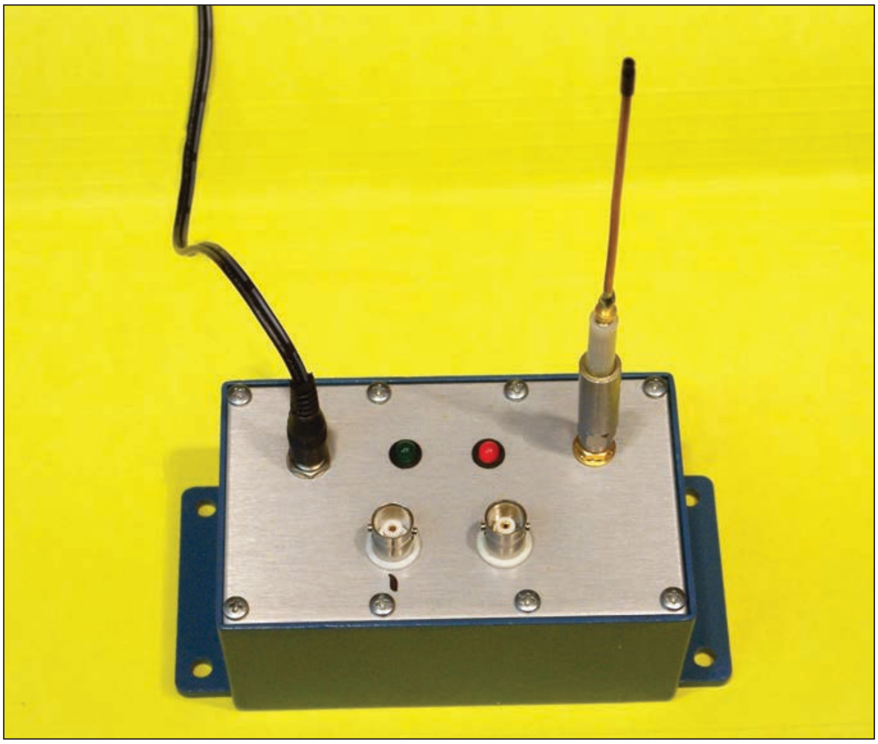


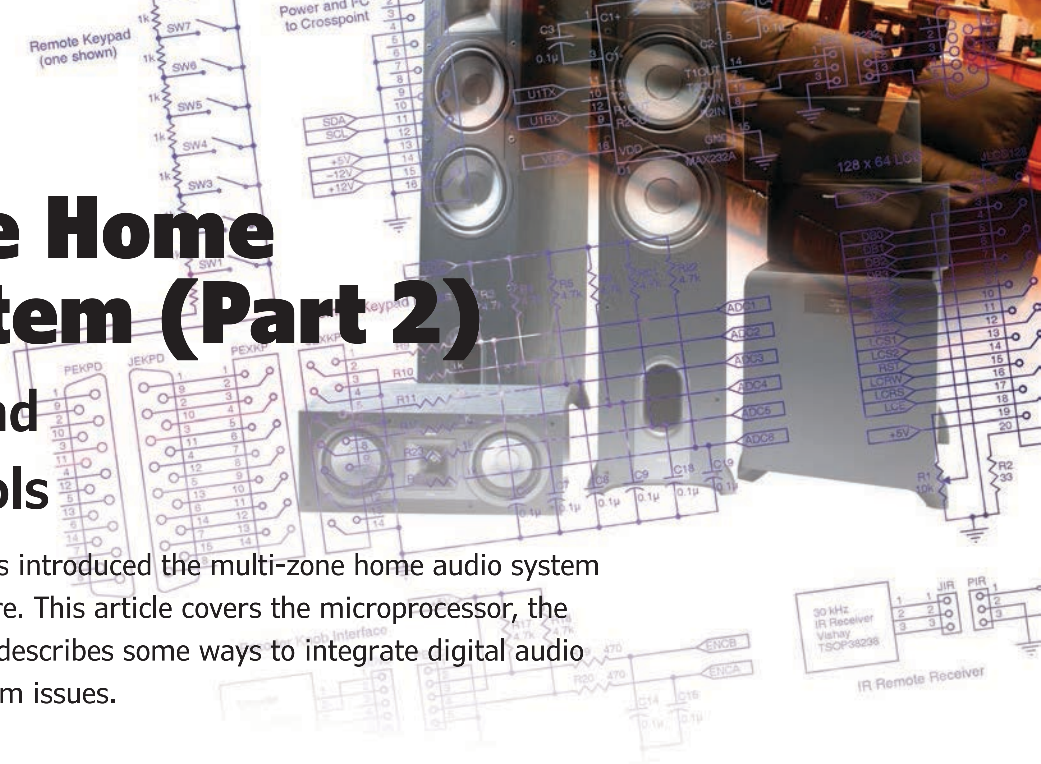
PHOTO 4
Here is my receiver circuit. One connector was used to monitor the signal strength voltage during development. The other connector feeds an input on a home alarm system. The short antenna reveals its 916.5-MHz operating frequency. Modules with a 433-MHz frequency will have a longer antenna.

to process the analog signal from the signal strength indicator (SSI). **Figure 4** shows the schematic. Note that the data pin doesn't function as you would expect. It isn't at a digital "1" level when the transmitter is on. It's only active when there's a serial datastream. The potentiometer sets the reception threshold. You can add whatever additional signaling circuitry you want for your application. Any rail-to-rail op-amp will work.

Your module's pinout may vary. The 916.5-MHz modules I used were surplus from another project, and they're no longer available from their source. However, 433 MHz is a more popular alarm frequency, so many inexpensive transmitter and receiver modules are available at that frequency. I suspect they have a similar construction, but ensure that they have an SSI pin. Or you must be willing to modify the source code to modulate the data and receive data at the transmitter and decode the data at the receiver. **Photo 4** shows my receiver, which is powered by a "wall wart" supply.

MODIFICATIONS

Although I designed the unit as a wireless alarm, it will work as an alarm without the wireless feature. Also, multiple interdigital sensors can be wired in parallel at a single alarm. That way, one alarm unit can monitor multiple water sources (e.g., a hot water tank, a clothes washer, and a home heating system boiler).



FEATURES

Multi-Zone Home Audio System (Part 2)

CPU, Controls, and Development Tools

The first part of this article series introduced the multi-zone home audio system and discussed the audio hardware. This article covers the microprocessor, the controls, and the display. It also describes some ways to integrate digital audio into the system and details system issues.

By Dave Erickson (US)

I used a microprocessor to control analog signal processing in my multi-zone home audio project. This approach is a good match for my capabilities and meets all of my goals. After all, analog is my favorite programming language.

A system could be set up digitally. In that case, the inputs would be either digital audio devices or, if they are analog, they would use an ADC for input. Then the source selection, volume, tone (filters), and other functions would be done digitally, either in firmware or by a DSP or FPGA. You could then send the sound digitally to the zones where a DAC or a "digital amplifier" could be used. Or why not distribute all the inputs to all the zones and make each zone a digital audio system?

DIGITAL VS. ANALOG AUDIO

A digital system would be outside my current skill set and would exceed my cost and complexity (design effort) budget. Ethernet is appealing for a wired system. Several pro-audio Audio over Ethernet (AoE) systems do this, but they are mostly proprietary and do not support Wi-Fi. Sonos is one commercial and proprietary Wi-Fi-based system designed for homes.

One AoE challenge is managing latency delays. Ethernet is not real time, so latency and thus delays are not controlled. Room-to-room delays of more than a few milliseconds would create objectionable echoes. I am not aware of any open solutions that offer delay matching.

One advantage of a centralized analog system is that the electronics can be located in one compact box. With analog, either speaker wires or line-level audio are run to each room. Wiring is simpler in a single-story home, but if you are an enterprising engineer who lives in a multi-story home, running a few wires probably won't stop you. With electrical, phone, cable, networking, sensors, audio, and so forth, I have a lot of wires in my house. It is simple to add a few more.

The multi-zone home audio system's cost per channel is roughly $120 to $150 per zone in an eight-zone system, including amplifiers and speakers. A zone that already has powered speakers, a PC, or a boombox costs about $60 since line-level audio can be wired to existing powered speakers. This cost does not include the labor of stuffing boards, loading code, and running wires.

Speakers can be just about anything including in-wall, in-ceiling, high-end, bookshelf, or indoor/outdoor types. I use a separate high-power amplifier and home-built speakers for my living room. **Figure 1** shows the system's design.

CPU AND CONTROLS

A home audio system needs several different but similarly functioning controls such as IR remote control to satisfy couch potatoes; front-panel controls for when the control is misplaced, or to do more complicated setups; remote zone controls to adjust the volume, source, or sound without having to return to the main system; and a display to view the status. Also, why not include a digital interface to enable system control from a PC? The challenge of all these controls is to make them act in a similar manner despite their very different electrical interfaces.

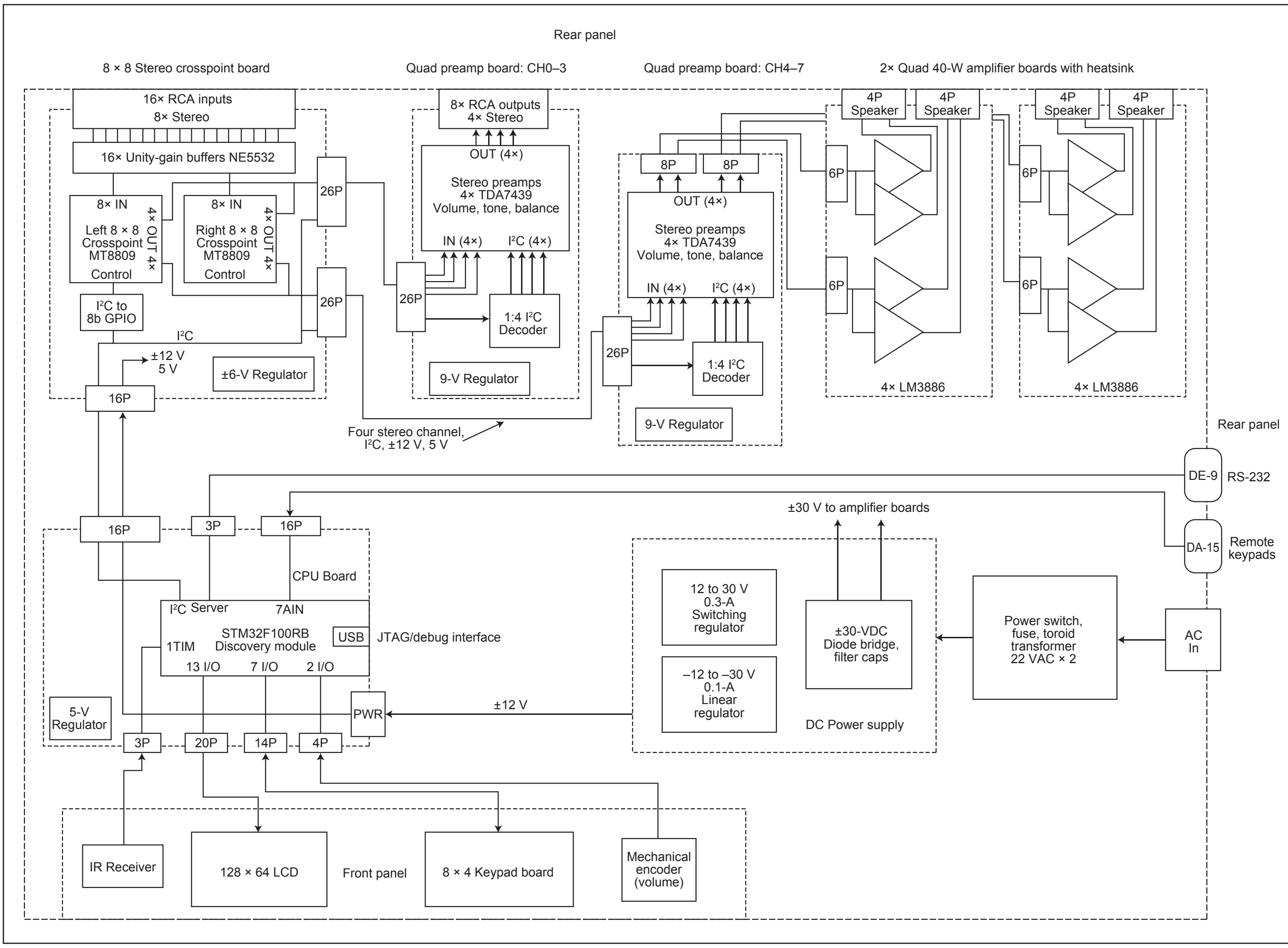


FIGURE 1
The system block diagram includes the boards, controls, amplifiers, and power supplies. The approximate component positions in the chassis are also shown.

GUI AND CONTROL STRATEGY

There are several ways to implement soft controls and displays. At one extreme, a multi-level menu system with soft keys can maximize the functionality of a few simple controls. On the other end of the spectrum, having one control per function enables you to perform functions without navigating menus.

Think of a calculator or an analog stereo with knobs or buttons for each basic function. Since it is important that my non-engineer family members can easily use the system, I used a graphic LCD combined with many labeled buttons for feedback.

Eight buttons select the eight zones, and eight more select the sources. Buttons for volume up and down, balance, bass, midrange, and treble are also included. There is a mute button as well as a mute-all button.

There aren't any multi-level menus yet. For example, to send the PC sound to the kitchen and adjust its volume, first select the zone, kit, then the source, PC, and then use the Volume Up and Volume Down buttons to adjust the volume.

Changes are displayed on the LCD in real time. This interface is fairly simple and intuitive. The buttons are arranged in a four-row by eight-column matrix. I color coded the buttons to help distinguish the functions. For control labeling, I used three 9-mm label tapes: one for the zones, one for the sources, and one for the other controls. I used arrays of ASCII strings for software labeling of the zones and the LCD's inputs. This required a recompile to change the house or input configuration.

Basic functions are repeated on the IR remote control and the remote keypads. Since these functions do not have visual feedback, controlling a remote zone could create problems as you continuously increase the volume but hear no response. Meanwhile, the neighbors four houses away can hear your deck speakers just fine.

So, to avoid trouble, I currently control only the local zone with the remote keypads. If you need to control a different zone, you need to get off the couch and go to the front panel. It would be amazing if the system had a smartphone app or at least a web browser interface.

FEATURES

CPU BOARD

Figure 2 shows the CPU board schematic. It contains an STMicroelectronics STM32VLDISCOVERY board as a daughtercard and interfaces for the LCD, analog remote keypads, the eight-row by four-column front-panel keypad, IR, the encoder knob, RS-232, and the I^{2}C connections for the preamplifier control. The CPU board is built as a hand-wired prototype but I plan to layout a PCB. **Photo 1** shows the front-panel setup.

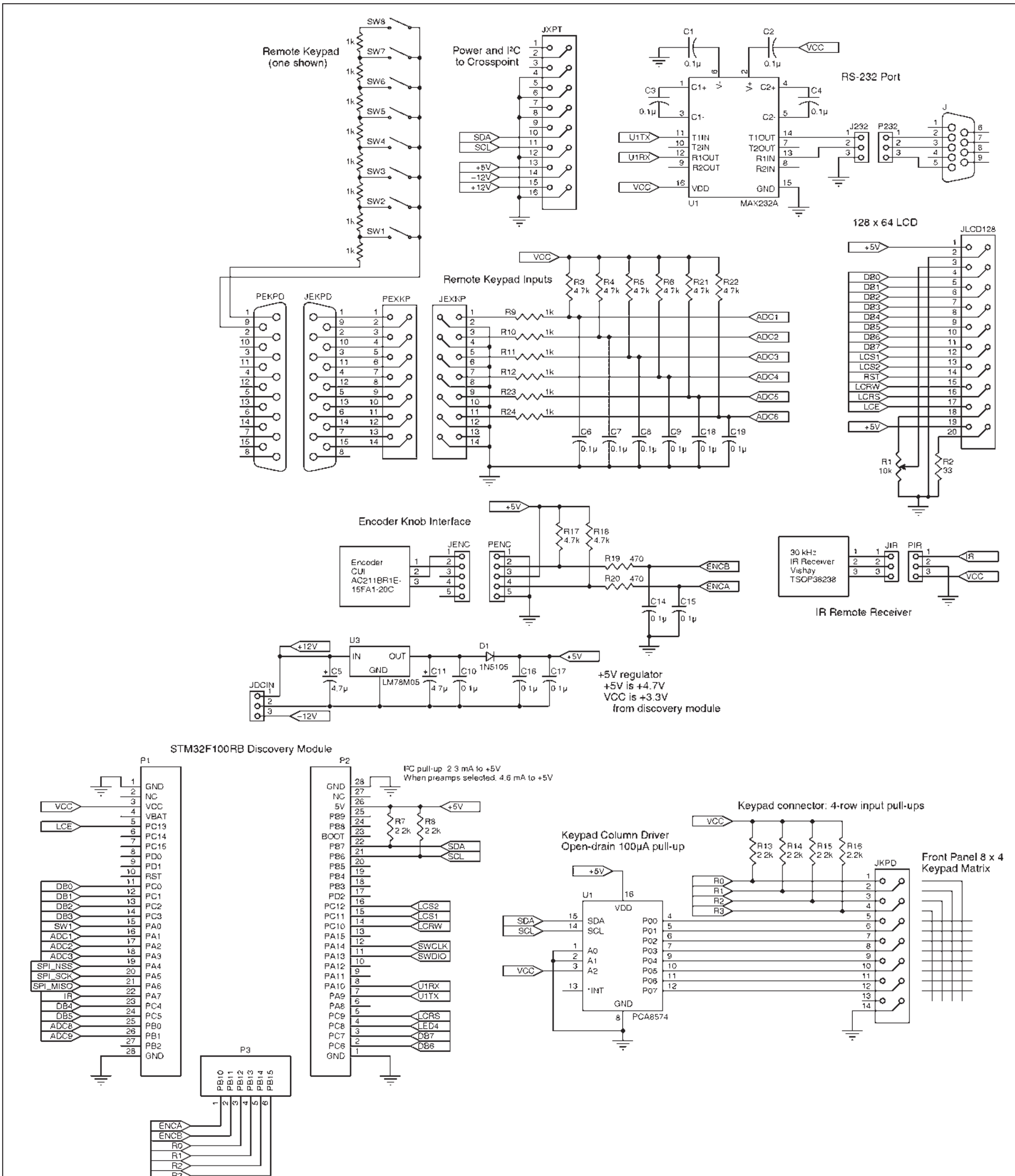


FIGURE 2
This is the CPU board schematic.

PHOTO 1
The front panel includes an LCD, keypad, and encoder knob.

FRONT-PANEL KEYPAD

The front-panel keypad is an ExpressPCB board containing 32 6-mm momentary buttons, organized as eight rows by four columns. The eight rows are driven by a PCA8571 I^2C GPIO port. Its outputs are open collector and pulled high, so pressing multiple keys causes no damage. The four columns are wired to processor input pins. This keypad is scanned one column at a time in a 1-ms timed-interrupt routine.

IR REMOTE

I used Sony remote control codes and emulated a Sony receiver. Sony codes use a simple protocol: a Start bit followed by 12 data bits. A Start bit is 2.4 µs wide, a Zero bit is 0.6 ms, and a One bit is 1.2 µs. The five most-significant bits (MSBs) of the 12 bits are the device code (CD, TV, DVD, etc.) and the seven least-significant bits (LSBs) are key codes.

The IR receivers detect the IR light, pass only the 38-kHz carrier frequency, and demodulate it to a CMOS logic signal. The firmware decodes these pulse widths and assembles the code.

If you haven't dealt with an IR decoder, it is all about error detection. Basically if you ever detect an error, toss out the code. By this, I mean reset a bit counter and then wait for the next Start bit.

The STM32VLDISCOVERY board's timers can directly measure pulse widths applied to input pins and generate an interrupt that informs you when a pulse has arrived. The next step is to read a register to get the pulse width and then compare the pulse widths to the minimum and maximum values for each of the Start, Zero, or one-pulse widths, which requires a maximum of six comparisons.

I clocked the 16-bit timer at 1 µs to provide plenty of resolution and to make the math convenient. A state machine waits in State 0 for a valid Start pulse to arrive. When it does, it counts the number of valid Zero or One bits and shifts them into a 16-bit value. If pretty much anything goes wrong, reset the state machine to a zero count. By "going wrong," I mean detecting any pulse that is not a valid one or zero for the next 12 pulses. When the count hits 13 (Start bit plus 12 data bits), you have a valid IR code. Well, almost.

For another test, count the number of identical IR codes in a row and only generate a valid key code when two identical codes arrive. IR remotes are intended to work this way, generating at least three identical codes in a row.

If you hold a button down, it continues to send the same code. Imagine a code being optically interrupted and then the end of another code arrives. Requiring two identical codes prevents this type of error.

Why do all this testing? We are surrounded by IR noise (e.g., from sunlight, incandescent and other lights, different remote controls, etc.). Also the IR signal can be weak, interrupted by the cat, and so forth. IR receivers contain a 38-kHz band-pass filter to help eliminate most ambient light, but this does not remove pulses from other IR remotes. All this may sound complicated, but is only about 20 lines of C and can be done in the timer interrupt handler. I have reliably used this method on this system and the original Freescale Semiconductor (formerly

FEATURES

LISTING 1
An array of these control data structures is used, one per control.

```
/* Controls structure: one per keypad. */
typedef struct {
  uint8_t  rawCode;       /* Key before LUT */
  uint8_t  rawFlag;       /* Key down from scan */
  uint8_t  keyCode;       /* Key after LUT */
  uint8_t  state;         /* State: 0-debounce 1-repeat dly 2-repeat*/
  uint8_t  sendFlag;      /* It's good. send it */
  uint16_t repeatTimer;   /* 1ms timer for key auto-repeat */
  uint8_t  repeat;        /* 1 if key should auto-repeat */
  uint8_t  zone;          /* Current zone for this keypad */
  uint16_t zoneTimer;     /* Time until this zone resets to default */
  uint8_t  zoneDefault;   /* Zone to default back to */
  uint8_t  oldKey;        /* Previous key for filter */
}inKeyTypeDef;
```

Motorola) 68HC11 microcontroller.

For a remote control, I used a One-For-All eight-channel universal remote to control all the equipment in my living room. These are low cost and readily available.

REMOTE KEYPADS

For each remote zone, an optional keypad provides basic controls including volume, source selection, balance, and muting. Remote keypads should be simple, inexpensive, and reliable. The microprocessor's eight-channel 8-bit ADC was unused on the original system.

An ADC is a terrible thing to waste, so I used it to read the remote keypads. Each keypad has eight momentary switches and a resistor ladder consisting of seven 1-kΩ resistors in series (press Button 0 and you get 0 kΩ, press Button 1 and you get 1 kΩ, etc.). In this way, up to eight remote keypads can be read, using one ADC channel per keypad.

Back on the CPU, a simple passive circuit is needed to read each channel's resistance. Each ADC input has a 4.7-kΩ pull-up resistor, a series 1-kΩ resistor, and a 0.1-µF filter capacitor. The 1 kΩ in series protects the analog pin in case of a static zap. Ideally, there should also be a protection diode on each input.

To detect a key press, the firmware periodically reads the ADCs, checks the voltage to see if any key is pressed, compares the voltage to the high and low thresholds for each of the eight keys, and increments a counter to debounce the keys.

I measured only one channel in the 1-ms timed interrupt routine. In fact, I triggered the ADC to measure the next channel after reading the current channel. This way, the interrupt routine doesn't need to wait for the ADC. Then the `key_proc()` code looks for 50 ms of the same keys detected to confirm a valid key press.

Keypad wiring uses a single unshielded twisted pair from the keypad back to the system. Polarity does not matter. However, running wires is inconvenient, so I am looking for a web interface to enable a PC or smartphone to control the system through a simple app or webpage. More on this follows.

ENCODER KNOB

Mechanical encoder knobs are inexpensive and provide a nice feel. Since volume control is important to this system, I decided to add a real volume knob. Like most encoders, the knob uses a two-bit quadrature code. There are many types of encoder signal timings. Some encoders output four 90° steps per mechanical click. The one I use outputs one 90° step per click.

The 1-ms interrupt routine reads the two encoder bits and combines them with the previous reading to build a simple 4-bit code from the two readings. Since this code contains both the current and previous states of the knob, a simple 4-bit (16-entry) look-up table (LUT) can determine what action to take.

The LUT has entries for Up, Down, or NOP. Using a LUT enables the code to be changed to accommodate various manufacturers' mechanical configurations. This LUT value determines whether a count value is incremented, decremented, or unaffected. The changes in the count can then be read in the main routine to change a setting (volume for now) based on whether the counter has increased or decreased.

MAKING CONTROLS CONFORM

Detecting a key press is simple. Debouncing and making the key periodically repeat after being held a while is a bit more complicated. Making 10 or so sets of controls comprising three totally different electrical types (front panel buttons, remote analog keypads, and an IR remote control) all act the same requires some thought.

Key auto repeat is useful for volume and other controls. To auto repeat, time how long the same key is pressed and, after a few hundred milliseconds, send another key code. Then wait a few hundred more milliseconds and resend it.

The IR remote presents a problem. Each remote generates codes at its own rate and repeats when the key is held down, but not necessarily at the repeat rate that you want. So a filter is needed that turns on when a key is pressed and times out after a period of time when the key is released. Then the IR buttons can be treated the same as any keypad.

Another issue is that any zone can be controlled by the front panel. But mostly you want to control the local zone from that zone. So, after a timeout period of no activity, each panel defaults back to the local zone.

I do not currently control any other zone from the remote keypads. When I do add this feature, these will have the same timeout mechanism.

If you combine all these requirements, the code can become complex. Fortunately, data structures are your friends. In **Listing 1** the struct `TypeDef` is used to control each keypad. An array of these structs is used, one for each keypad.

In addition to the multiple control sources, during system debugging it is beneficial to use a PC keyboard to control the system and a PC display to output debug `printf()` messages via a terminal emulator and RS-232. I used a single keyboard ASCII key to emulate each control.

For example, “V” is volume up and “v” is volume down. The right inputs are selected by “1” through “8.” The eight zones are selected by “SHIFT_1” through “SHIFT_8.” It is helpful to choose commands you can remember. I use “h” for Help to display a list of the commands.

A big `case` statement interprets and executes all the commands and uses these single ASCII codes as its selector. The other

```
Set X address (x & 0x3F)
     Set bus direction OUT
     Set RW, RS, both CS
     Set data
     Pulse E
E= 1, Delay 500ns, E = 0
     Delay(5us)
Set Y page (y >> 3)
     Set data
     Pulse E
     Delay(3us)
Pixel mask (1 >> y%7)
Read the dummy data
     Set bus direction IN
     Set RW, RS
     Select L or R chip based on X>63
     Pulse E
     Delay(5us)
Read the real data
     E = 1
     Delay(1us)
     Input data
     E = 0
     Delay(3us)
Reset X address (x & 0x3F) since the last read incremented it
     Set bus direction OUT
     Set RW, RS, both CS
     Set data
     Pulse E
     Delay(3us)
Merge the mask and the read data:  (mask | data)
Write data
     Set bus direction OUT
     Set RW, RS
     Select L or R chip based on X>63
     Pulse E
     Delay(3us)
```

LISTING 2
This is the pseudocode for `putpix()` to write a single pixel to the display.

various controls use a small LUT to map their binary outputs into the same ASCII codes. I invented this simple method long ago and suspect that many others have also come up with it.

A nice side effect of this technique is that your project has the foundation of a serial protocol to control it remotely. If you add a serial to USB chip, then presto! Your project has a USB interface.

GRAPHIC LCD: THE UBIQUITOUS KS0108

The goals for a front-panel display are to show all of the parameters of one zone at a time and to graphically and interactively show the volume, balance, and tone settings. The LCD should be small, but not too small; highly visible in any lighting; and low cost. I like the look of white LED backlights. I decided to use a 128 × 64 monochrome panel based on the KS0108 controller chip. These low-cost chips are readily available from several manufacturers. I chose the NHD-12864WG-BTFH-V from Digi-Key, which costs about $20.

"A home audio system needs several different but similarly functioning controls. The challenge of all these controls is to make them act in a similar manner despite their very different electrical interfaces."

The hardware interface is straightforward. It has eight data bits and five control signals. The KS0108A uses two devices; each accesses one half of the display, so two chip selects are needed.

One of the first tasks of this project was to write the LCD hardware access-level code. I found a few examples online for KS0108 code and borrowed a few ideas, but finally decided to write my own.

I have a higher-level LCD library, which I have used with my own FPGA-LCD based controller designs since the 1990s (see my article "Graphics LCD Control for Embedded Applications," *Circuit Cellar* 34, 1993). Using an off-the-shelf panel with its built-in controller was a real experience, since these chips have a few quirks. They are byte-oriented and the pixels within a byte are vertically organized. Since most font and bitmap files are organized with horizontal bytes, they would need to be transposed either by the microprocessor or before they are loaded into code. Fortunately, I found a 5 × 8 font designed for the KS108A. I haven't needed bitmaps yet, so I haven't had to face that challenge.

GRAPHICS FUNCTIONS

To make good use of a graphics LCD, you need functions to draw common objects. Lines, filled rectangles, and ellipses (circles) require a pixel draw routine. Characters and bitmaps need a byte write function.

Display clearing and updating should be fast enough that you don't notice flicker between the time that you clear the display and the time that you write the new data. It should be just a few milliseconds to avoid flicker. The fast 24-MHz ARM Cortex-M3 processor can do the job without a lot of special optimizations. I used a 1-MIPS 8-bit 68HC11 microcontroller and a larger LCD in this project's previous version, so the code had to jump through hoops to update the LCD fast enough to prevent flicker. The FPGA-based LCD controller helped since it was designed to make drawing primitives fast enough with even a slow processor.

The KS0108 vertical-byte data organization requires that fonts are eight pixels high including spaces. Otherwise, you would need to write them one pixel at a time, which would be quite slow. So the only practical small font is 5 × 7. Having your characters placed anywhere vertically except on a byte-boundary is pretty painful so I accept this limitation.

Larger fonts fit into two or more bytes, but I don't currently use these. One approach I have used to generate larger characters is to pixel replicate the small font by two or three times. This has the advantage of using only one small font table but the disadvantage that larger fonts can appear blocky.

I like to render lines and circles in the LCD memory, so I needed to write graphics one pixel at a time. Sounds simple, right? Unfortunately with the KS0108 such a seemingly simple operation requires an inordinate amount of code. **Listing 2** shows the pseudocode for a `putpix(x, y)` function just to write one pixel.

Part of the complexity is the large number of delays needed to access this older slow device and to meet all its setup and hold timings. A pixel write requires you to do a read/modify/write to the memory and single byte reads are inefficient. To do a read, first a dummy read is required. Then, since the X address always auto increments, the address needs to be set back again before the real read. I used a 24-MHz processor with plenty of code space, but the delays alone add up to about 25 µs. I estimate about 40-µs total CPU time per pixel. I have not measured it.

One thing that the KS0108 does reasonably well is to move a block of data from CPU memory to the LCD. That is because the addresses increment automatically after an access. So writing byte-aligned fonts is fairly fast. Another approach to manage a display is to render the entire display in CPU memory

much more quickly and then move the entire screen to the LCD (128 × 64/8 is only 1 KB). But for larger panels, this approach uses a lot more CPU memory and more time to update the screen. For example, a 320 × 240 1-bit panel needs 9,600 memory bytes, so it would require about 10 times as long to move that data. Rendering graphics in the display minimizes CPU memory use.

Currently only two screens are displayed: a startup "splash" screen that displays the code revision and date and a single-zone status screen. I plan to add at least one more screen, using large characters, so I can read the source and zone from across a room.

DEVELOPMENT TOOLS

The STM32VLDISCOVERY board modules are a great deal. They offer a DIP module that brings out every pin of the 64-pin processor and the ST-LINK USB debug interface, which is STMicroelectronics's two-wire JTAG programming and debug interface, all for $12.

STMicroelectronics's 2011 STM32 Design Challenge offered a free starter version fAtollic's C development tools. Unfortunately, when the contest ended, Atollic imposed a 32-KB code size limit in its free version. Since my code was already 49 KB and growing, this was a problem.

After looking at Yagarto and other toolsets, I found CooCox's CoIDE, an integration of Eclipse, GCC, GDB, a lot of programmer and debugger support, and nearly every ARM CPU manufacturer's device libraries, all free and without limitations. CooCox's website has many user-supplied examples. Porting to CooCox was fairly painless, as is changing to other ARM devices.

ARM PERIPHERALS

I am used to the peripherals on 8-bit devices, most recently the Atmel AVR microcontroller. I was surprised at the extensive features and complexity of STMicroelectronics's ARM Cortex-M3 devices.

For example, all the GPIO ports are 16 bits and each bit can be controlled several ways by multiple registers. In addition to writing all 16 output bits or their direction register, there are multiple-bit Set and Clear registers for both the data and the direction registers. These enable multiple device handlers to access the same port's individual bits without interference. On typical 8-bit processors, care must be taken to prevent interference between multiple device handlers.

There are several of each type of peripherals. On this mid-end processor, there are multiple ADCs, DACs, I^2Cs, UARTs, SPIs, timers, and so forth. Using an I/O register's name directly is not a practical way to handle multiple devices since too many unique names would be required.

STMicroelectronics offers extensive libraries to help write device code. At first I was intimidated by the dozens of functions just to access an I^2C device, for example. However, once you figure out what functions you need to do your job, the rest goes smoothly.

Even device initialization can be daunting when there are dozens of registers with possibly hundreds of bits. To help, ARM uses a configuration data structure and provides an initialization function such as `DAC_Configuration(void)`. First you set the structure elements, then you call the function to transfer the struct to the device. All devices can be initialized in this way.

DIGITAL AUDIO SOURCES

In our home, we currently use a standard desktop PC in the den as a music server to play our MP3 collection via Winamp and to stream audio from Pandora or other services. For a local MP3 player, a simple mini-plug to RCA cable will do the job, but it does not

PROJECT FILES

circuitcellar.com/ccmaterials

RESOURCES

Atollic Inc., www.atollic.com.

D. Erickson, "Graphics LCD Control for Embedded Applications" *Circuit Cellar*, 34, 1993.

Digi-Key Corp., www.digikey.com.

ExpressPCB, www.expresspcb.com

Raspberry Pi, www.raspberrypi.org

Sonos, Inc., Sonos system of hi-fi wireless speakers and audio components, www.sonos.com

Yagarto, Yet Another GNU ARM Toolchain, www.yagarto.org

SOURCES

ARM Cortex-M3 processor
ARM, Ltd. | www.arm.com

AVR Microcontroller
Atmel Corp. | www.atmel.com

BeagleBone Black
BeagleBoard.org | www.beagleboard.org.

CoIDE Software development environment
CooCox | www.coocox.org

68HC11 Microcontroller
Freescale Semiconductor, Inc. (formerly Motorola) | www.freescale.com

STM32VLDISCOVERY board and ST-LINK in-circuit debugger/programmer
STMicroelectronics | www.st.com

ABOUT THE AUTHOR

Dave Erickson (dave@djerickson.com) has been an electronics hobbyist since the 1960s. He earned his BSEE in 1976. Dave worked at HP Medical, Datacube, Analogic, Zoll Medical, Teradyne, and numerous startups. He currently develops electro-optic and ultrasound systems for a cardiac catheter system at Infraredx. Dave's electronics interests include instrumentation, audio, electronic music, and boat electronics. He also enjoys biking and sailing. His projects are available at www.djerickson.com.

provide either charging or a stand. iPod docking stations with audio outputs can be purchased for that purpose.

GROUNDING AND HUM

Most consumer audio gear is not power line grounded and uses two-wire line cords. The various RCA audio connections provide a local "ground reference" and things work well enough. Connecting to a single grounded audio source doesn't generally create a ground loop or hum problem. Connecting two or more grounded sources creates a ground loop and, depending on the power line ground-voltage difference, can causes varying amounts of hum.

I chose to ground my system since it is the center for many audio signals throughout our house. Connecting to ungrounded equipment doesn't present a problem, but connecting to other grounded equipment can cause hum.

Examples of grounded audio components are PCs and most cable TV boxes. A cable box's RF cable is grounded where it enters the house for lightning protection. If you connect your PC or cable box to an audio system and don't get hum, great. If there is hum, commercial audio isolation transformers will solve the problem.

FUTURE FEATURES

One feature on my wish list is the ability to control the system from a web page. This would enable a smartphone or PC anywhere in the house to control the system. One reason I used a powerful ARM processor when a lesser CPU would probably do was to have the resources to someday serve up webpages and handle TCP/IP.

You may ask, "So where is the webpage, Dave?" The truth is that I do not currently possess the skills to generate active webpages and handle file systems, TCP/IP stacks, and so forth.

I have seen projects that implement simple web servers on an 8-bit processor and I consider these interesting, but that is all. They typically do not have a real server, a file system (for images, HTML, etc.), or file management tools. They use a lot of `sprintf()` commands to render HTML or JavaScript on the fly. They typically do not have DHCP or system configuration tools.

If you open a port to the Internet, you will need to deal with security issues. After using high-level tools and real servers to write web pages, this approach seemed primitive.

I like to spend my hobby time developing skills and systems that are applicable to my career or at least to a real commercial product. I tend to avoid developing toys, tricks, and hack code. So as I wait patiently for someone to drop a nice web server, file system, and TCP/IP stack for STM32 processors in my lap, time marches by.

At this point, I am leaning toward using a low-cost, low-power Linux board (e.g., a Raspberry Pi or BeagleBoard.org's BeagleBone Black) to address these features. They support Apache and other web servers with Internet security, full local and network file systems, music (and video), and servers (e.g., XBMC), all written and supported by serious programmers. And it all comes in a credit-card size $50 board that consumes a watt or so.

A web browser, Pandora, or any other streaming web function is just a download away. I have not worked much with Linux on embedded controllers, but it is a skill I would like to develop. So many projects, so little time.

BUILDING YOUR OWN

I used ExpressPCB to design the boards. If there is interest, I will offer bare PC boards on my website. The boards are designed with surface-mount technology (SMT) electronics, except for the connectors and the film capacitors, which are through-hole. The SMT parts are mostly 0.05" pitch and 0805 or larger and can be built under a magnifier by hand-soldering. I used ribbon cables where possible to minimize cable assembly labor. The CPU board is currently hand-wired. The ExpressPCB layout in available on the project website.

This has been a rewarding project for me. If you are interested, the full ExpressPCB schematics, PCB artworks, BOMs, and code are available on *Circuit Cellar*'s FTP site. The project website is available at www.djerickson.com/multizone.

FEATURES

The Home Energy Gateway

Remotely Control and Monitor Household Devices

The Home Energy Gateway system enables users to monitor energy consumption and remotely control household devices. An embedded gateway/web server communicates with a smart meter, which monitors average active power consumption and several smart plugs in a home area wireless network. A user can monitor the smart plugs and use a web interface to adjust them.

By Raul Alvarez Torrico (Bolivia)

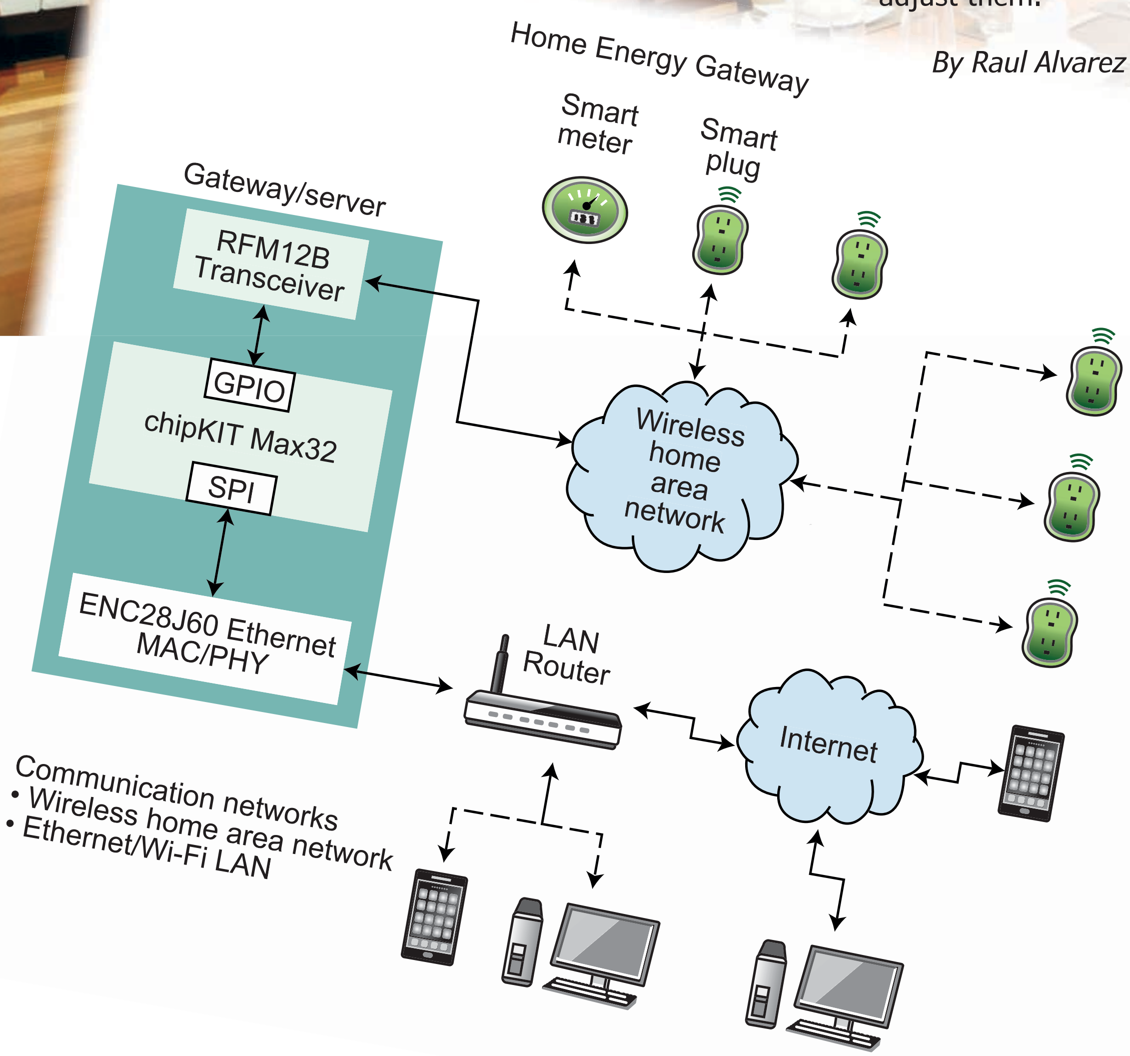


FIGURE 1
The Home Energy Gateway includes a Hope Microelectronics RFM12B transceiver, a Digilent chipKIT Max32 board, and a Microchip Technology ENC28J60 Ethernet controller chip.

I have been always amazed by the Internet's potential and how this technology provides so many possibilities for human interaction. The Internet enables server and client computers to share data from one side of the world to the other! So, when the idea of the Internet of Things (IoT) came around, it was like rediscovering the Internet.

Since I graduated from college, I have been very interested in communication protocols, especially for wireless networks. These networks seem more "mysterious" and they are generally more complex due to the inherent unreliability of wireless data transmissions. This project puts into practice what I have been reading and learning about those topics in recent years.

This article describes the design and implementation of my Home Energy Gateway prototype, a system that won second place in the 2012 DesignSpark chipKIT challenge. The system enables users to remotely monitor their home's power consumption and control household devices.

The main system consists of an embedded gateway/web server that, aside from its ability to communicate over the Internet, is also capable of local communications over a home area wireless network. This wireless network includes one smart meter that measures the house's power consumption and several smart plugs to which household devices (e.g., fans, lights, coffee machines, etc.) are attached. The gateway is connected to the house's LAN and can be accessed locally or remotely over the Internet. When accessing the gateway/web server over the LAN or Internet, the user sees a web interface containing the controls to turn on/off the smart plugs and sees the monitored power consumption data that comes from the smart meter in real time.

As a learning exercise, I specifically developed the communication protocol I used in the home area wireless network from scratch. I used low-cost RF transceivers to implement the protocol. It is simple and provides just the core functionality necessary for the application.

BLOCK DIAGRAM AND FUNCTIONAL DESCRIPTION

Figure 1 shows the system's block diagram and functional configuration. The smart meter collects the entire house's power consumption information and sends that data every time it is requested by the gateway. In turn, the smart plugs receive commands from the gateway to turn on/off the household devices attached to them. This happens every time the user turns on/off the controls in the web control panel.

I used the simple wireless protocol (SWP) I developed for this project for all of the home area wireless network's wireless communications. I used low-cost Hope Microelectronics 433-/868-/915-MHz RFM12B transceivers to implement the smart nodes.

The wireless network is configured to work in a star topology. The gateway assumes the role of a central coordinator or master node and the smart devices act as end devices or slave nodes that react to requests sent by the master node.

The gateway/server is implemented in hardware around a Digilent chipKIT Max32 board. It uses an RFM12B transceiver to connect to the home area wireless network and a Microchip Technology ENC28J60 chip module to connect to the LAN using Ethernet.

As the name implies, the gateway makes it possible to access the home area wireless network over the LAN or even remotely over the Internet. So, the smart devices are easily accessible from a PC, tablet, or smartphone using just a web browser. To achieve this, the gateway implements the SWP for wireless communications and simultaneously uses Microchip Technology's TCP/IP Stack to work as a web server.

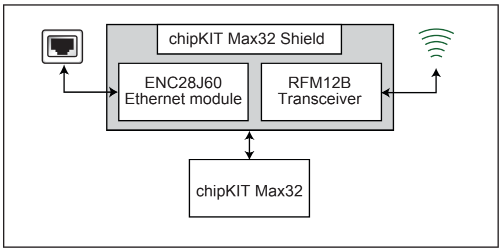


FIGURE 2
The gateway hardware module is shown.

Thus, the Home Energy Gateway generates and serves the control panel web page over HTTP (this page contains the individual controls to turn on/off each smart plug and at the same time shows the power consumption in the house in real-time). It also uses the wireless network to pass control data from the user to the smart plugs and to read power consumption data from the smart meter.

I can't describe all the project details in this article, so I'll concentrate on discussing the project's wireless and TCP/IP communication aspects and explaining how they functionally interface with each other. I won't include minor details about the sensing, metering, and actuation; low-level information about the hardware/software; and the SWP design and implementation specifics. The schematics, source code, and support material are available on *Circuit Cellar*'s FTP site (see Project Files).

HARDWARE SYSTEM MODULES

The entire system's hardware can be divided into three main modules: the gateway module, the smart meter module, and the smart plug modules.

The Gateway Hardware Module—As **Figure 2** shows, the gateway module comprises three main sub modules: The chipKIT Max32 board, the RFM12B wireless transceiver, and the ENC28J60 Ethernet module. The wireless transceiver and the Ethernet module are both connected to a custom "Shield" board (Arduino Mega style) for the chipKIT Max32.

The chipKIT Max32 is a powerful microcontroller board that is based on Microchip Technology's PIC32MX795F512L high-end 32-bit microcontroller. This MIPS architecture-based processor has 512 KB of flash memory and 128 KB of RAM. It can operate up to 80 MHz and has an array of peripherals and communication modules,

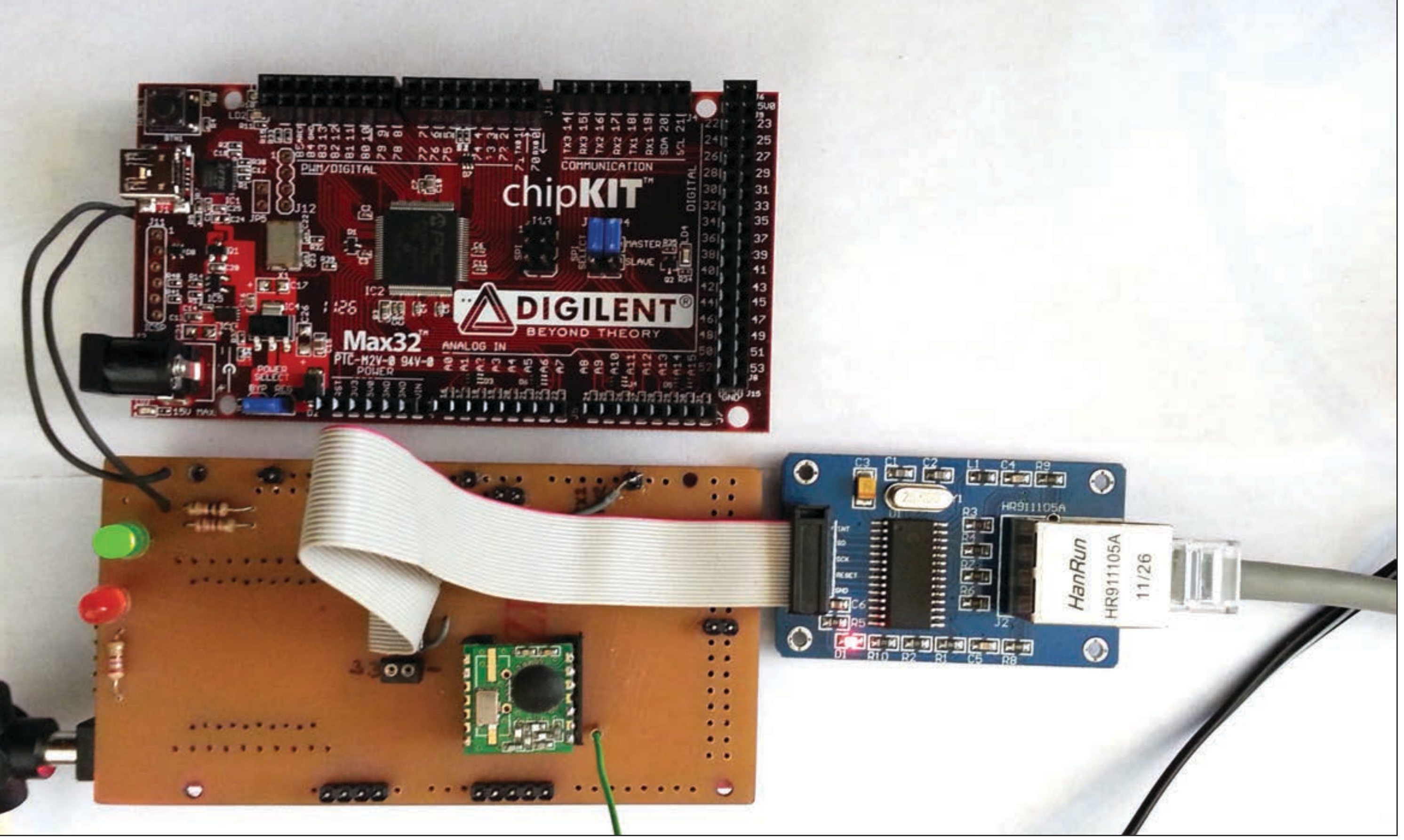


PHOTO 1
The Home Energy Gateway's hardware includes a Digilent chipKIT Max32 board and a custom shield board.

including a 10/100 Ethernet MAC.

The RFM12B is a low-cost FSK transceiver with available models operating in the 433-/868-/915-MHz ISM bands. I used the 433-MHz version for this project. This transceiver includes data rates up to 115.2 Kbps, programmable transmission frequency deviation, programmable reception bandwidth, automatic frequency control, signal strength indicator, sync pattern recognition, and a SPI, among other things.

The Ethernet ENC28J60 module I used is a generic module built around Microchip Technology's ENC28J60 MAC/PHY chip. It uses a flat cable to physically connect to a 5 × 2 header in the shield board. Although the PIC32MX795F512L has an Ethernet MAC (but not a PHY) and a Digilent Network Shield for the chipKIT Max32 is available, I decided to use this module because it was what I had on hand at the time. Nevertheless, it is also possible and relatively easy to configure the TCP/IP stack to work with the Microchip Technology (formerly SMSC) LAN8720 chip present in the Network Shield.

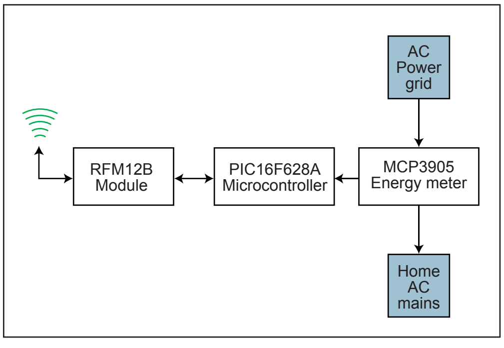


FIGURE 3
The smart meter hardware module is shown.

Photo 1 shows the gateway hardware prototype. The custom shield's circuit diagram is not included in this article but is available at *Circuit Cellar*'s FTP site (see Project Files).

The Smart Meter Hardware Module—**Figure 3** shows a diagram for this module's hardware configuration. It has an RFM12B transceiver for wireless communications as well and uses a low-cost, low-end 8-bit Microchip Technology PIC16F628A microcontroller as a main processor. The attached power metering module is a Microchip Technology's MCP3905RD-PM1 Reference Design Board for the MCP3905 energy-metering IC. Using this board simplifies the power metering task because it has all the necessary circuitry in place along with the MCP3905 chip to do the measurements. It also provides a digital pulse output with a variable frequency that is proportional to the measured average power consumption. To obtain the power consumption data, you only need to count the pulses over time.

As you may already know, the PIC16F628A isn't a very capable microcontroller. I selected it to prove you could use a small microcontroller to implement a lightweight wireless protocol. This microcontroller can achieve minimum functionality for a wireless monitoring and control application. It can also simultaneously verify the minimum required memory footprint for the design and determine necessary coding in C language.

When I designed the protocol and wrote the code, I was in the process of learning about the subject (I still am), so my protocol design and coding approaches may not be the best. In the end, choosing a relatively tiny microcontroller and starting to code a wireless protocol with it was more a matter of a personal challenge than making the best design decision. (It gave me some headaches, but I had a lot of fun doing it!)

The protocol is still a work in progress and obviously this PIC doesn't have the necessary resources to support enhancements to the protocol's functionality. But the code can be easily ported to another more powerful microcontroller without significant changes. I already had to port it to PIC32 architecture for the gateway module and it didn't give me much trouble.

The PIC16F628A runs on its 4-MHz internal oscillator and uses pins RB3 through RB7 for SPI communications with the transceiver. The code uses bit banging for the SPI because this microcontroller doesn't have a SPI module.

The MCP3905 energy metering board has an optocoupled digital pulse output, which provides the necessary isolation to the microcontroller and the rest of the circuit from the AC grid. This digital pulse output is fed to the PIC's RB0/INT input to detect the signal's frequency. This frequency is typically orders of magnitude lower than the microcontroller's clock, so the microcontroller's RB0/INT pin is configured to work as an external interrupt pin to efficiently count the low-frequency signal's rising edges. This count data is sent to the gateway every time it is requested. As I mentioned earlier, this signal's frequency is proportional to the measured average power consumption and the board has to be calibrated first to provide accurate data.

I won't discuss further details about the energy meter board's hardware. The user's manual, which explains the circuit and includes a detailed description of its design and operation, is available for download (see Resources).

Photo 2 shows three smart node hardware prototypes. **Photo 2a** shows a smart plug prototype, **Photo 2b** shows a second smart plug in a breadboard, and **Photo 2c** shows the smart meter prototype.

The Smart Plug Hardware Module—**Figure 4** shows the smart plugs' main hardware components. The microcontroller and radio transceiver are both the same as for the smart meter, but the smart plugs also have a Sharp Microelectronics S212S01F solid-state relay to turn on/off the household devices. This relay has a 12-A maximum output current and it works with voltages up to 600 VAC_P, so a wide range of household devices can be connected to it. The PIC16F628A controls the relay through digital output pin RA0.

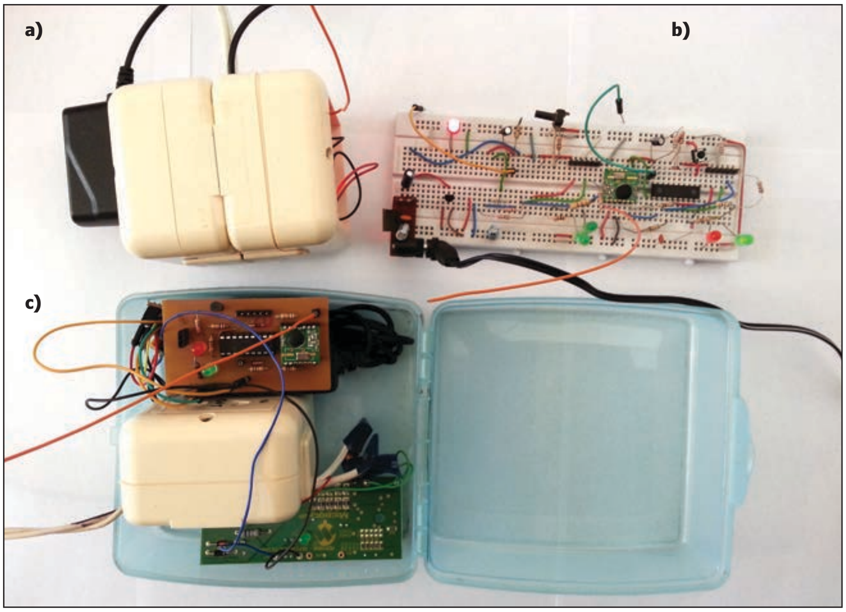


PHOTO 2
These are the three smart node hardware prototypes: a smart plug **(a)**, a second smart plug in a breadboard **(b)**, and the smart meter **(c)**.

Note that it is possible to combine both smart plug and smart meter functionalities in one. In fact, that's what I did with my prototypes. So, I'm providing just one circuit diagram that combines the hardware required for both types of nodes (the differences between them are minimal anyway). The code and wireless protocol also enables this combined functionality. This makes it possible to use hybrid smart plug meter nodes to control a household device and take readings of its power consumption at the same time. Nevertheless, I found it more convenient to explain both modules separately in this article for the sake of clarity.

Figure 5 shows the circuit diagram of a node that can work as a smart plug, a smart meter, or a hybrid.

SOFTWARE SYSTEM MODULES

The entire system's software can also be divided into three modules: The gateway firmware, the smart meter firmware, and the smart plug firmware.

The Gateway Firmware—The gateway's firmware is written in C language for the Microchip Technology C32 compiler. **Figure 6** shows its submodules. The gateway

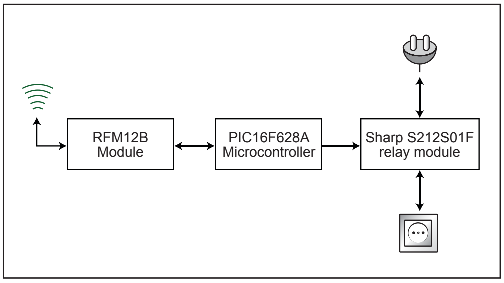
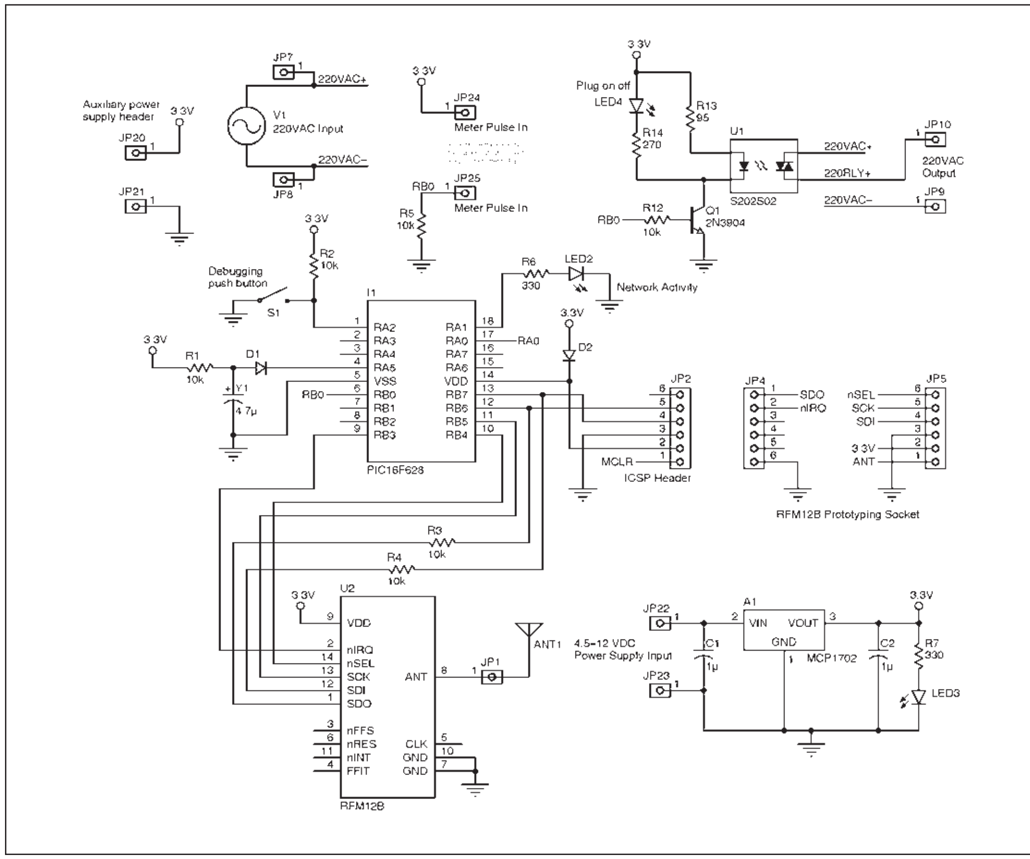


FIGURE 4
This is the smart plug hardware module.

FEATURES

FIGURE 5
This is the smart plug/meter circuit diagram. The node works as a smart plug, a smart meter, or a combination.



application runs on top of the Microchip Technology TCP/IP stack and the SWP. The TCP/IP libraries make it fairly easy to implement the HTTP web server functionality, which provides an HTML web control panel with AJAX technology to generate automatic information updates without reloading the page. **Photo 3** shows a screen capture of the web control panel.

Access to the page is authenticated and a list of one or more user accounts can be statically preconfigured in the gateway/server's firmware. There is a preconfigured account in the provided source code. The user name is "admin" and the password is: "ccellar."

For those willing to take a look into the gateway's code, a good starting point would be the "CustomHTTPApp.c" file, which contains the functions to interface with the web server application protocols (DDNS, HTTP, SNMP, SMTP, etc.) and callback functions to interface with the smart nodes in the custom application. I have added all the necessary code to interface the application tasks with the TCP/IP code in this file. I used one of the available project examples that came with the TCP/IP stack as a starting point. That made it relatively simple to blend my code with the web server implementation.

Another worthy file is "MainDemo.c", which contains the `main()` function that is obviously the entry point for the entire gateway code. I also had to add my own code in this file. I declared some variables for my custom application, called some initialization functions, and added the necessary routines to interface with the wireless protocol and monitoring/control tasks. The particular project I chose from the available TCP/IP stack examples required that one must adhere to a cooperative multitasking mechanism, in which custom application tasks must run to completion and return as soon as possible.

The Simple Wireless Protocol—The SWP and driver for the RFM12B transceiver were first developed for the PIC16F628A microcontroller. The code was then ported with

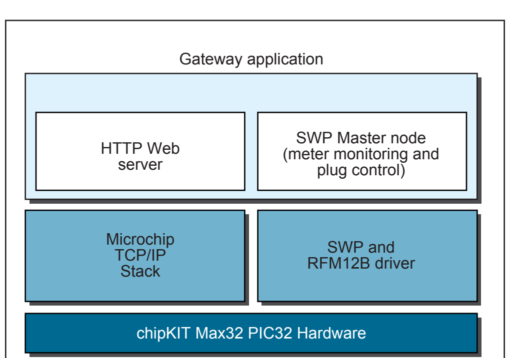


FIGURE 6
The gateway firmware module is shown.

minor modifications to the PIC32MX795F512L architecture.

As the name implies, the SWP is very simple. It comprises just the necessary functionality to send and receive packets using simplified PHY and MAC layers.

To give you an idea of the code size, the last compilation for the 8-bit PIC16F628A microcontroller gave me 1,855 bytes of program memory and 181 bytes of RAM, including the smart devices' application code (which I compiled using Microchip Technology's HI-TECH C compiler, optimizations enabled). The protocol implements the basic functionality for peer-to-peer communications and star topology network and is inspired to some extent by the IEEE 802.15.4 (2003) specification.

The RFM12B transceiver's driver code is almost the same for both microcontroller architectures; however, the PIC32MX795F512L's code utilizes an external interrupt pin to interface with the interrupt out signal provided by the transceiver. This signal is generated every time the transceiver's internal reception buffer is filled and must be immediately read. However, for the PIC16F628A I'm not using an external interrupt pin for the same purpose. I am only using a regular input pin and software polling because, apparently, the capability of running a function from within the interrupt routine is very limited in this microcontroller, perhaps due to stack constraints.

The protocol's physical layer provides the minimum functionality for the following tasks: initializing the transceiver's operation frequency over a list of 20 predefined channel frequencies, setting and changing the transmission output power, sending/receiving data frames, and performing clear channel assessments (the last is still being beta tested). The physical layer also provides the upper MAC layer with an interface for driver initialization, reading and writing frames, setting and receiving addresses, and generating or performing a CRC on MAC packets.

On the other hand, the MAC layer provides the functionality for sending and receiving packets between pairs of nodes located in each other's transmission/reception range (routing capability is not implemented). At the time of this writing, the MAC layer supports transmission and reception of MAC data and MAC commands, with automatic acknowledge of received packets and automatic retransmissions in case of failure. The retransmission takes place several times (predefined in code) when the source node doesn't receive a MAC acknowledge from the destination node.

The MAC layer also provides to the upper layers functions for reading and writing data packets, a packet-received indication data structure, and access to MAC parameters (e.g., source and destination addresses, frame control, transmission/reception status, etc.). All data to be passed between layers is implemented in code as structs accessed via pointers.

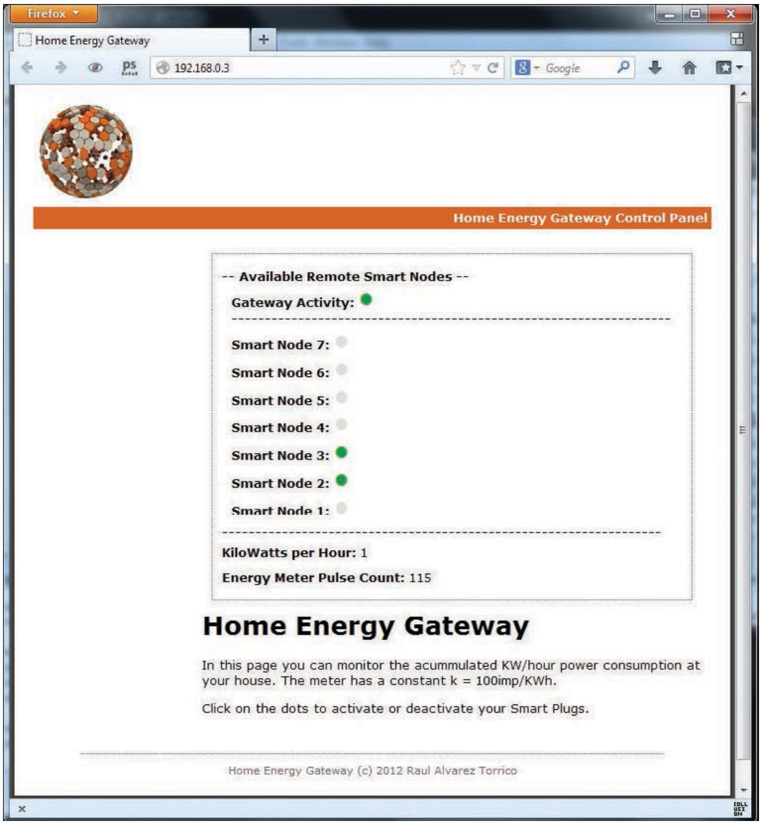
Although this SWP prototype worked well, it has yet to be thoroughly tested. I think it will be relatively easy to improve the implementation and functionality over time.

FEATURES

ABOUT THE AUTHOR

Raul Alvarez Torrico (raul@tecbolivia.com) has a BE in Electronics Engineering and is currently a freelance engineer. In his spare time, he likes to experiment with wireless sensor networks, robotics, and artificial intelligence. He is also committed to publishing articles and video tutorials about embedded systems and programming in his native language (Spanish). More information is available at his website (www.tecbolivia.com).

PHOTO 3
Multiple users can be given access to the web control panel.



The Smart Meter Firmware—The smart meter's PIC16F628A code is written in C language for the HI-TECH C compiler. **Figure 7** shows the firmware sub modules for the plug and meter nodes. As I mentioned earlier, it is possible to have a node simultaneously working as a plug and a meter. The wireless protocol and application code is written to enable the operation of both profiles without much hassle.

In the case of the smart meter, basically the microcontroller counts the pulses received from the power-metering board and sends this data every time the gateway requests it (e.g., every 5 to 10 s, defined in the firmware). Persistence of this data is not implemented in the meter node or in the gateway. I left this feature to be implemented in the future because this first prototype mainly concentrates on the communication tasks and network interfacing.

The SWP code and RFM12B transceiver's driver code are essentially the same for the smart meter, the smart plug, and the gateway with minor modifications pertaining to both PIC architectures. The smart meter receives `CMD_REQ_DATA` application-level commands from the gateway as a request to send power consumption data obtained from the meter circuit.

The Smart Plug Firmware—Nearly the same details and explanation I just provided for the smart meter also apply to the smart plug. The software (and even the hardware) for both types of nodes are basically the same.

The smart plugs receive `CMD_REQ_ACTION` application-level commands from the gateway and act accordingly, activating or deactivating the relays and hence, the particular loads (household devices) connected to them.

LOOKING TO THE FUTURE

The Home Energy Gateway is a very capable low-cost system for DIY home automation enthusiasts. The use of low-cost PIC16F628A microcontrollers and RFM12B transceivers, added to the PIC32 microcontroller's computational power, made this an interesting project to experiment, learn, and have fun with home automation.

Needless to say, the system has a lot

PROJECT FILES

circuitcellar.com/ccmaterials

RESOURCES

Arduino, http://arduino.cc.

Microchip Technology, Inc., "MCP3905/6 Evaluation Board User's Guide," 2005.

SOURCES

chipKIT Max32 board and chipKIT Network Shield
Digilent, Inc. | www.digilentinc.com

RFM12B Transceiver
Hope Microelectronics Co., Ltd. | www.hoperf.com

PIC32MX795F512L and PIC16F628A Microcontrollers, ENC28J60 Ethernet controller chip, LAN8720 Ethernet transceiver, MCP3905RD-PM1 Reference design board, MCP3905 energy-metering IC, C32 and HI-TECH C compilers, and MPLAB IDE
Microchip Technology, Inc. | www.microchip.com

S212S01F Solid-state relay
Sharp Microelectronics of the Americas | www.sharpsma.com

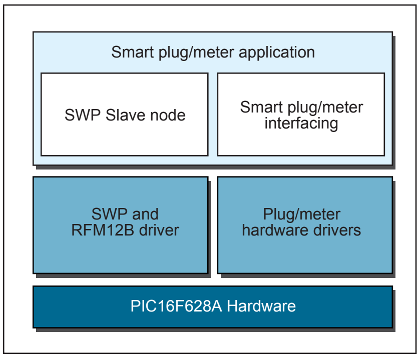


FIGURE 7
The smart plug/meter firmware module enables a node to function as a plug and a meter.

of room for improvement. Although the hardware and software at a prototype level showed great potential and worked well, the current solution is far from being complete or professional grade.

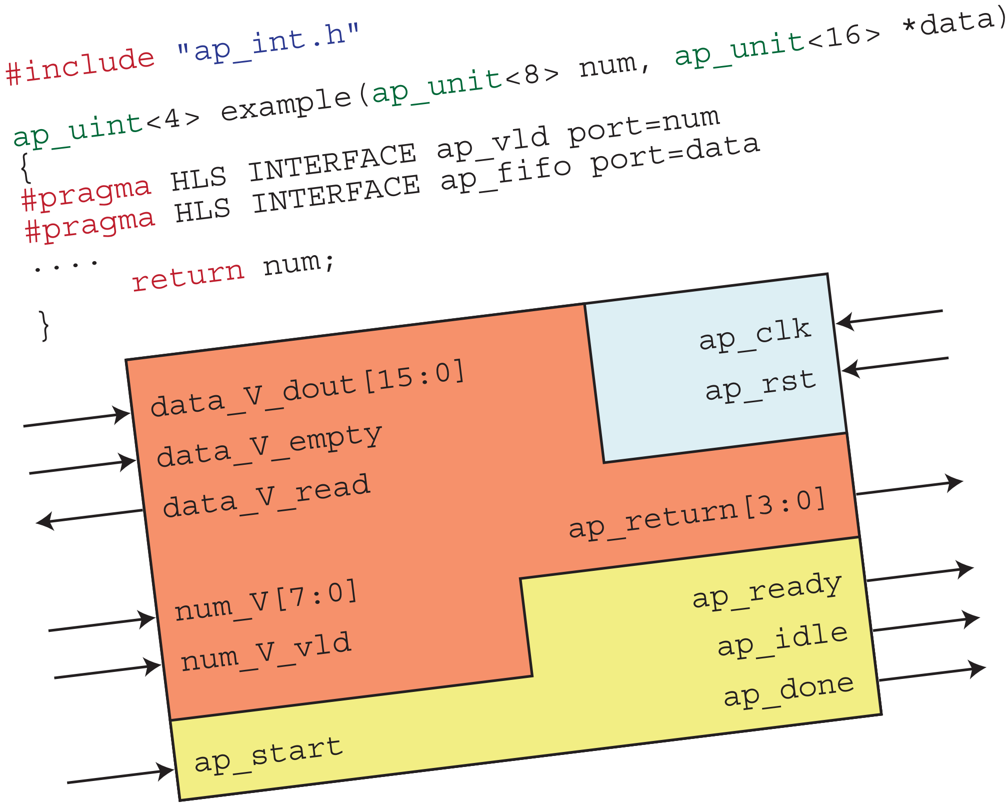
I would like to address some improvements to the project in the future. For example, the SWP doesn't implement security and encryption. Implementing them would require the use of more powerful microcontrollers in the smart nodes, which definitely must be implemented in future versions of the system. Also, the protocol design must be thoroughly tested and the functionality offered at the PHY and MAC Layers must be improved. I may even expand their capabilities with an additional upper network layer for automatic network formation, routing, and more versatile network topology capabilities.

In future versions of the system, I would like to include the ability to send notifications and reports to the user via e-mail. This could be done by implementing a small database in the gateway to store a user's power consumption and add device control profiles.

I would also like to add the following: analog or continuous control in the nodes for devices that require more than binary on/off type of control signals, ZigBee capability to the gateway for communication with ZigBee nodes, home safety and security functionality using other types of sensors and actuators, and a hardware control panel with an LCD and keypad as an alternative user interface.

All circuit diagrams, additional pictures, and the Microchip Technology MPLAB IDE projects for the gateway and nodes are available for download (see Project Files). I really enjoyed working on this project and I am also grateful to have had the opportunity to share some details of it with you through this writing. Feel free to send me your comments and suggestions.

PROGRAMMABLE LOGIC IN PRACTICE



Rapid FPGA Design in C Using High-Level Synthesis

It usually takes more effort to implement an FPGA design than to execute a similar microprocessor design. This article features high-level synthesis examples to show how you can use the C/C++ design language to simplify your FPGA design.

By Colin O'Flynn (Canada)

FIGURE 1
An example of mapping a C function into an FPGA block is shown above. Several of the ports are standard for each block (e.g., the `ap_start` and `ap_done`). Different interface types generate different control signals: `ap_vld`, and `ap_fifo` are shown here. The `ap_vld` interface has a single line that tells the block when valid data is present. The `ap_fifo` interface can be used to consume many bytes of data, for example, data that would come from a first in, first out (FIFO) buffer.

It's no secret that it's harder to implement an algorithm in an FPGA compared with a software-only solution. To combat this, various "high-level" design tools have tried to provide a method to help you get a solution up and working quickly. I'd like to introduce you to Xilinx's Vivado High-Level Synthesis (HLS) tool, which is designed to use C as an FPGA design language.

I'm excited about the Vivado HLS for many reasons—perhaps most importantly because it's now easy to get your hands on it for experimenting. While the HLS tool isn't included in Vivado's free (WebPack) edition, you can easily obtain a free 30-day trial license if you want to follow along with this article. Otherwise, the tool is available separately for about $2,000 or as part of the Vivado System Edition.

Before I begin, I want to make something clear: The HLS shouldn't be used to translate *software* written in C into FPGA designs. Fundamentally you are designing hardware. If you use these tools with a hardware-designing mindset, you will have a successful and enjoyable experience. If you try to ram C code written for a processor through the tools, you will have a terrible experience.

Likewise, the tools won't make a good FPGA designer out of a software engineer; you still need someone with FPGA experience. Xilinx has attempted to ease this transition a little with User Guide UG998, "Introduction to FPGA Design with Vivado High-Level Synthesis," a document introducing HLS and FPGAs. It provides a great start and a good overview, but you still need the hardware experience for the best results.

TOOL SYSTEMS

The Vivado HLS's basic idea is that you can use C/C++/SystemC to design your FPGA or FPGA blocks, which alone should make your life easier in the design phase. But there is another advantage. You can actually simulate your design by just compiling the C/C++ code and running it on your computer. This gives you a huge speed improvement over running a Verilog/VHDL simulation.

You may want to refer to User Guide UG902, "Vivado Design Suite User Guide High-Level Synthesis," which is Xilinx's documentation for the HLS tool. The UG902 document is around 600 pages, so I've tried to link to some specific areas of interest from ProgrammableLogicInPractice.com. You can


```
#include "ap_int.h"
#define FIR_LENGTH 19
typedef ap_fixed<12,2> fir_t;

void fir(volatile ap_uint<10> * x, volatile ap_uint<10> * y)
{
#pragma HLS INTERFACE ap_fifo depth=19 port=x
#pragma HLS INTERFACE ap_fifo depth=19 port=y
  fir_t b_k[FIR_LENGTH] = {-0.000859, -0.000229, -0.001539,
  -0.015266, -0.012840, 0.060680, 0.150479, 0.087929, -0.138514,
   0.727273, -0.138514, 0.087929, 0.150479, 0.060680, -0.012840,
  -0.015266, -0.001539,-0.000229,-0.000859};

  fir_t sum;
  ap_fixed<17,7> intsum;
  ap_int<12> temp;
  int i;

  fir_t x_store[FIR_LENGTH];

  while(1){
#pragma HLS PIPELINE II=1
    //Update input data
    LOOP_UPDATE: for (i=0; i < FIR_LENGTH-1; i++){
      x_store[i] = x_store[i+1];
    }

    //Zero-Shift from ADC
    temp = (ap_int<12>)*(x++) - (ap_int<12>)512;

    //Map to ap_fixed Variable
    x_store[FIR_LENGTH-1](11,0) = temp(11,0);

    intsum = 0;
    LOOP_FIR: for (i=0; i < FIR_LENGTH; i++) {
      //Multiply & Add
      intsum += b_k[i] * x_store[(FIR_LENGTH-1)-i];
    }

    //Map to ap_uint variable
    temp = fir_t(intsum).range(11,0);

    //Zero-Shift for DAC
    *(y++) = (ap_uint<10>)(temp + (ap_int<12>)512);
  }
}
```

LISTING 1
This is the complete implementation of a FIR filter in C++. The input and output are served by a pair of 10-bit ADC/DACs, which use an offset-binary format. Data is internally mapped into 12-bit fixed-point numbers, onto which the FIR filter algorithm is applied.

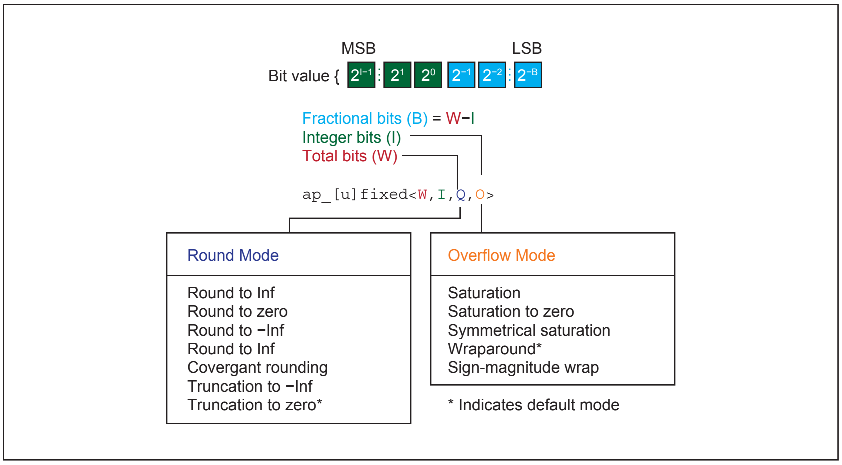


FIGURE 2
You can use the `ap_fixed<>` data type in C++ programs to provide you with fixed-point math. Since everything is bit-accurate, you can validate and test the effect of fixed-point math on your algorithm by simply implementing the algorithm in C++. In addition, there are many more options (e.g., rounding/truncation) than simple fixed-point implementations.

also look at new features in the Vivado release notes. These tools are still rapidly evolving, so it's worth keeping your eyes open.

I'll be using C and C++ for the examples in this article. Both C and C++ give you "arbitrary precision" integer types, so you can specify *any* number of bits in the integer, not just the usual 8, 16, 32, or 64 bits. For example, you can have a 33- or 2-bit integer type. C++ also includes some additional features (e.g., fixed-point number support).

If you've only used C before but want these features, don't worry. You can simply rename your source file .cpp instead of .c. All C code is valid as C++, so you can ignore all the other C++ features you don't need to use.

When designing a project, you can specify a testbench written in C/C++. Part of the synthesis project can automatically run your testbench and validate the design works as intended. Again, because you are running the testbench on the C/C++ code and not as part of a Verilog simulation, it often takes only a few seconds to run.

BLOCK INTERFACE

How do you conceptually map a C function into an FPGA block? The basic idea is accomplished via special directives that tell the synthesis tool what control ports you want around each C function argument. **Figure 1** shows some examples.

You can think of each C function as being

```
uint16 smartcard(uint8 cla,     /* APDU Header: CLA Byte */
                              ...
                    volatile uint8 * uart_out,  /* To UART, streaming */
                    volatile uint8 * uart_in    /* From UART, streaming */
                    )
{
#pragma HLS PROTOCOL fixed
#pragma HLS INTERFACE ap_fifo depth=32 port=uart_in
#pragma HLS INTERFACE ap_fifo depth=32 port=uart_out
...
  //Output header, first five bytes
  *(uart_out++) = cla;
  *(uart_out++) = ins;
  *(uart_out++) = p1;
  *(uart_out++) = p2;
  *(uart_out++) = p3;
  ap_wait();

  //Read one-byte input
  procedure_byte = *(uart_in++);
  if (procedure_byte == ins){
    *status = 1;
  } else {
    *status = 0;
  }
...
  ap_wait();

  //Read expected response (if any)
  resp = *(uart_in++) << 8;
  resp |= *(uart_in++);

  return resp;
}
```

LISTING 2
Some detail of the smartcard interface block designed with the HLS in C is shown. Note the special `ap_wait()` command is used along with the `'#pragma HLS protocol'` directive to enforce the correct order of the read/write first in, first out (FIFO) buffer.

```
static uint8_t state[16];
static uint8_t statekey[16];

void aestest(ap_uint<128> *inptext, ap_uint<128> *key, ap_uint<128> *outtext)
{
#pragma HLS RESOURCE variable=return core=AXI4LiteS metadata="-bus_bundle aes_io"
#pragma HLS RESOURCE variable=inptext core=AXI4LiteS metadata="-bus_bundle aes_io"
#pragma HLS RESOURCE variable=outtext core=AXI4LiteS metadata="-bus_bundle aes_io"
#pragma HLS RESOURCE variable=key core=AXI4LiteS metadata="-bus_bundle aes_io"
#pragma HLS ARRAY_PARTITION variable=state complete dim=1
#pragma HLS ARRAY_PARTITION variable=statekey complete dim=1
#pragma HLS PIPELINE
#pragma HLS inline recursive
...
}
```

LISTING 3
This interface and the Xilinx High-Level Synthesis (HLS) directives are used to implement a high-speed Advanced Encryption Standard (AES) coprocessor in C++ for the Zynq device.

turned into a hardware block. The `ap_start` port tells the block to begin running. When it's finished, you get a signal on the `ap_done` port. This type of operation would be similar to presenting all the data to the function, running the algorithm, and getting a final output. But you can also create functions that simply process data.

FINITE IMPULSE RESPONSE EXAMPLE

My first example, which is a finite impulse response (FIR) filter, will show this type of function. Its main body has an infinite `while(1){}` loop. The function never returns, it simply consumes input data and generates output. (Xilinx's User Guide UG902 includes a list of all possible interface port types with specific details.)

Listing 1 shows the C++ code for the implementation. The filter's "b" values (coefficients) define how it reacts. You can design a low-pass, band-pass, high-pass, or band-stop by simply changing those coefficients.

These coefficients are real numbers that have values such as −0.03245 or 0.08621. This means when you implement the design, you cannot just use simple integer math. You need to use either floating- or fixed-point representations. Fixed-point numbers mean the bits used to hold the number are split into an "integer" portion (i.e., the portion of the number to the left of the decimal point) and a "fractional" portion (i.e., the portion of the number to the right of the decimal point). These bits provide a specific range of values the number can represent and always have the same granularity.

When using fixed-point numbers, you need to ensure you allow enough bits for the maximum size of the number and decide what will happen if the number overflows. There are also trade-offs about how many bits to use to represent the fractional portion. More bits mean better granularity, but a larger final design.

I could easily fill several articles with my thoughts about using fixed-point numbers—and getting it wrong can cause "issues." For example, see the case of the Ariane 5 rocket, which failed (at a cost of half a billion dollars)

```
void init_table(int16_t sin_table[256])
{
   int i;
   for (i = 0; i < 256; i++) {
      double real_val = sin(M_PI * (double)(i - 128) / 256.0);
      sin_table[i] = (int16_t)(32768.0 * real_val);
   }
}

int32_t array_ROM_sin(int16_t inval, uint8_t idx)
{
   int16_t sin_table[256];
   init_table(sin_table);
   return (int32_t)inval * (int32_t)sin_table[idx];
}
```

LISTING 4
The `init_table()` routine uses floating-point math and a call to the `sin()` function, all written in C. But the HLS tools will optimize this into a look-up table instead, which helps reduce complexity in your design.

Elektor.STORE

The world of electronics at your fingertips!

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

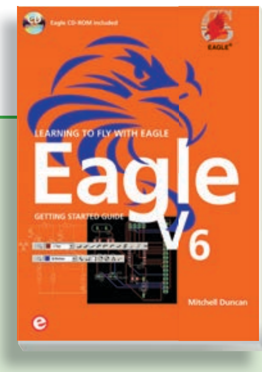
Books

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 systematical 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. While actively working on these projects the reader gets to truly comprehend and master the basics of the underlying controller technology.

260 pages • ISBN 978-1-907920-25-7 • $56.40



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 • $47.60

Programming step-by-step

Android Apps

This book is an introduction to programming apps for Android devices. The operation of the Android system is explained in a step by step way, aiming to show how personal applications can be programmed. A wide variety of applications is presented based on a solid number of hands-on examples, covering anything from simple math programs, reading sensors and GPS data, right up to programming for advanced Internet applications. Besides writing applications in the Java programming language, this book also explains how apps can be programmed using Javascript or PHP scripts.

244 pages • ISBN 978-1-907920-15-8 • $56.40

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

Books

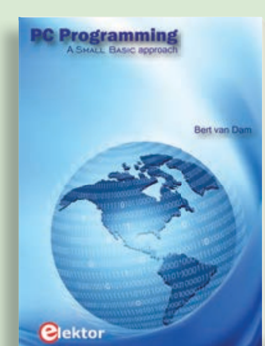
Bestseller!

Very Original Applications!

Mastering Microcontrollers
Helped by Arduino

Arduino boards have taken the world by storm since their initial development. The aim of this book is not only to let you enter the World of Arduino, but also to help you emerge victorious and continue your microcontroller programming learning experience by yourself. In this book theory is put into practice on an Arduino (uno) board using the Arduino programming environment. Having completed this fun and playful course, you will be able to program any microcontroller, tackling and mastering I/O, memory, interrupts, communication (serial, I²C, SPI, 1-wire, SMBus), A/D converter, and more.

348 pages • ISBN 978-1-907920-23-3 • $56.40



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 • $47.60

Q-Watt Audio Power Amplifier

Elektor has a long history with audio power amplifiers. We are proud to present yet another fully analog circuit developed entirely in house. Despite the simple design of this audio power amp with just one pair of transistors in the output stage, Q-Watt can deliver over 200 quality (Q!) watts into 4 ohms with exceptionally low distortion thanks to the use of a special audio driver IC. Due to popular demand, this project is now supported by a semi-kit containing everything you need to build a stereo amplifier.

2 PCBs and all electronic parts

Art.#110656-72 • $244.00

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

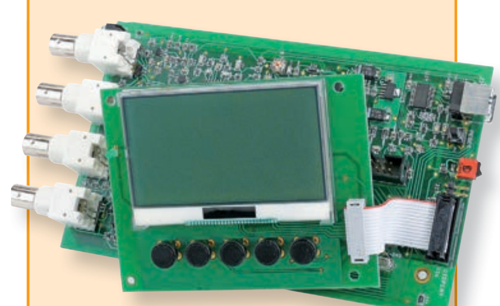
Kits & Modules

Android Elektorcardioscope

Instructive, fascinating, and potentially useful to everyone: perform your own electrocardiograms on your Android smartphone or tablet! The project involves skillfully combining a small PIC interface to control an analog input stage with a great deal of software. Our ECG interface is available in the form of a ready-to-use module to which you just have to add four electrodes and an Android application for smartphone or tablet; there's no physical connection between this terminal and the interface, as it uses Bluetooth communication!

Ready assembled board

Art.# 120107-91 • $132.00



500 ppm LCR Meter

The remarkable precision of this device and its amazing ease of use are the result of careful design. It works so well behind its uncluttered front panel that one could almost forget the subtleties of the measurement techniques employed. A dream opportunity for our readers who are passionate about measurement to enjoy themselves. If, like us, you wonder at the marvels modern techniques bring within our reach, come along and feel the tiny fraction of a volt.

Set: Main board and LCD board, assembled and tested

Art.# 110758-93 • $300.00

Optimization Added	Clock Cycles	BRAMs	LUTs
None (initial AES source)	1,369	5 (1%)	646 (1%)
#pragma ARRAY_PARTITION state, statekey complete	670	3 (1%)	1,299 (2%)
#pragma HLS PIPELINE	121	2 (<1%)	1,256 (2%)
Duplicate SBOX look-up table	51	20 (7%)	1,208 (2%)
#pragma HLS inline recursive	5	20 (7%)	7,122 (13%)

TABLE 1
This table shows an implementation of an Advanced Encryption Standard (AES) core on the ZedBoard, which has a Xilinx Zynq FPGA. By simply varying the directives, you can easily switch between area or speed optimizations with the same source code.

due to an incorrect conversion from floating-point to integer representation. Once again, I cannot stress enough that these tools cannot be used without understanding what you are really asking them to do.

The HLS tools provide a special C++ class giving you access to a robust and simple-to-use fixed-point number class (see **Figure 2**). Remember all these features are bit-accurate when you compile your code for the C++ simulation (i.e., you can easily see the effect of different bit-length, rounding, or overflow parameters).

When implementing an algorithm, it's a good idea to use a `typedef` to define a new type for your variable (e.g., the FIR coefficient type). You may wish to first use a float type and validate the code is working as expected in the C++ simulation. Then, by changing the `typedef`, you can switch into the fixed-point version of the code.

One additional feature that the HLS with C++ gives you is special "streaming" interfaces, which are designed to enable simulation of data interfaces (e.g., on my example FIR filter). See the Designing with Streaming Data section of User Guide UG902 for more information about streams.

TALK THAT TALK

Here is a common problem I run into that HLS makes easy: I need to implement a fairly complex state machine, often to handle talking to some peripheral device (e.g., an I²C sensor). Such a task is better suited to a microcontroller. One solution might be to implement a small soft-core processor in your design. But this brings with it additional work, and unless I have other work for the soft-core processor to do, it seems like overkill.

In this particular example I'll implement a module to control a smartcard. Two standard UART blocks are used to connect to the smartcard. Again, I'll use the first in, first out (FIFO) interface to connect my HLS code to the other blocks. The UART blocks provide a FIFO-like interface that can be easily connected to the module resulting from HLS synthesis. Like many "protocol"-type designs, this code needs to send some bytes, wait for a response, then act on that response.

Listing 2 is an example of the code, which is a portion of the complete source. In the beginning it sends five bytes, receives a status byte, and checks if the received byte is the correct value. A special call to `ap_wait` (for C) or `wait` (for C++) *must* be inserted between switching between Write and Read mode in this example. Also, the entire section must be declared with the `#pragma HLS PROTOCOL` option.

Without this special setup, the synthesis tool may rearrange the order of read/writes to I/O ports. This means it may try to read from `uart_in` before writing the five bytes out. From the synthesis perspective, this makes perfect sense: nothing in the code indicates dependency between the two variables even though they are `volatile`. It can also reduce latency by scheduling the read to occur before the writes are finished. But the resulting design won't work, because when it goes to read from the input FIFO it will block until data is present. Since data will only become present once the five bytes are written to the output, the design will stall.

The C/C++ testbench probably won't catch this problem, since you'll have preloaded data into the input FIFO and then validated the output FIFO has the expected data. Be vigilant when dealing with ports that have some sort of external dependency on order, such as in the example I provided.

BUILDING A PERIPHERAL

So far I've been using all simple FIFO interfaces and connecting them into an

circuitcellar.com/ccmaterials

RESOURCES

ProgrammableLogicIn Practice.com.

Xilinx, Inc., "Introduction to FPGA Design with Vivado High-Level Synthesis," User Guide UG998, 2013.

———, "Vivado Design Suite User Guide High-Level Synthesis," User Guide UG902, 2012.

SOURCES

Cortex-9 processor
ARM, Ltd. | www.arm.com

Vivado High-Level Synthesis (HLS) tool, Zynq programmable SoC, Xilinx Platform Studio (XPS), Xilinx Software Development Kit (SDK), AXI4 IP, and AXI4-Lite IP interface
Xilinx, Inc. | www.xilinx.com

ZedBoard development kit
ZedBoard | www.zedboard.org

ABOUT THE AUTHOR

Colin O'Flynn (coflynn@newae.com) has been building and breaking electronic devices for many years, and 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 his prototype circuits. Some of his work is posted on his website at www.newae.com.

existing FPGA design. For my final example I'll use a ZedBoard development kit to design a custom encryption peripheral that connects to a hardcore ARM Cortex-9 processor inside a Xilinx Zynq programmable system on a chip (SoC). In future articles, I will provide more details about the ZedBoard platform and Zynq device, but for now all you need to know is it combines an ARM processor with some FPGA fabric.

For this example I won't touch any Verilog or VHDL code, but will still manage to do a fairly complex task: implement an encryption core as a custom processor peripheral and drive it from a program compiled onto that processor. I'll also demonstrate how HLS makes it fairly trivial to switch between different optimization goals.

I'll use 128-bit operations to design an Advanced Encryption Standard (AES) core. I need to use a standard interface to connect this block to the processor. In this case, my peripheral needs a Xilinx AXI4 bus interface. From within HLS, this simply requires me to specify the AXI4-Lite interface type. The HLS tools also contain an option to export the design so I can include it in Xilinx Platform Studio (XPS) as a peripheral block, and a software driver I can pull into the Xilinx Software Development Kit (SDK) to compile a simple example.

Full details of the AES algorithm are beyond the scope of this article, but plenty of examples and simple C source codes are available online. (I've also included links at ProgrammableLogicInPractice.com.) For this example, I've simply taken the most straightforward C implementation you'd be likely to find and passed it through the synthesis tools. I haven't applied any specific hardware implementation tricks, which can further tune performance (e.g., replacing look-up tables with efficient hardware calculations).

Listing 3 shows the top-level AES function. **Table 1** shows the resulting performance and area usage for the core with various options enabled. It's worthwhile to note that not only can the resulting design have very high performance, but you can easily tune the resulting synthesized core for area/speed trade-offs.

In this example, I used the `ARRAY_PARTITION` directive to split some large arrays into individual elements instead of using BRAMs. Be careful about overusing this directive. For example, if you attempt to split-up static arrays which can be implemented as ROMs (as in the sbox variable in the source), it may result in extremely long implementation time without any performance increase. Visit the companion website to see the full AES design along with instructions to connect it to the hardcore processor.

ROM DESIGN

When implementing algorithms in an FPGA, certain functions may be moved into look-up tables. Standard functions such as `sin()`, `cos()`, and `log()` could be moved to look-up tables. But any custom equation could also be implemented in a look-up table, the size of the table depends on the number of bits in the data representation.

With HLS you can integrate the "generator" function into your code and the HLS tools will automatically generate an appropriate ROM (look-up table) instead of synthesizing your generator function (see **Listing 4**). Integrating the Generator function into your code greatly simplifies problems such as needing to recreate the table when parameters (e.g., number of bits in your fixed-point representation) change.

IMPLEMENTATION COMPLETE

The HLS tool is a powerful method to design FPGA systems (or subsystems) without mucking too much in very low-level details of the implementation. This article has only scratched the surface of how to use these tools. Xilinx has many resources you can follow for more information. As always, the full source code for all the examples is available at ProgrammableLogicInPractice.com. I also posted demonstrations of the implemented design running in an FPGA and videos of using the tools to enable you to follow along without the commitment of downloading and installing the entire Vivado HLS software package.

EMBEDDED IN THIN SLICES

Embedded File Systems (Part 3)

Designing Robust Flash Memory Systems

How can you make a system last with something as complex as a Linux flash memory system? This article details solutions for extending a flash memory system's life.

By Bob Japenga (US)

My house has too much stuff—especially electronic junk. I have various electronic gadgets that I just don't throw away. I still have the hard drive from my 1990 PC. It has 10 MB of data on it! Just recently, I was about to throw away—I mean, recycle—a PCMCIA Wi-Fi card. The built-in Wi-Fi on my wife's laptop recently stopped working. In 5 min I was able to get it working from my stash of electronic "junk." See, I am not a pack rat. I am a supply chain warehouse!

Part of the reason I have so much electronic junk is that as a culture we don't keep electronics very long. How long have you had your cell phone? How long did you keep your last cell phone? How about your DVD player? Did you replace it with a Blu-ray player? Those industries don't have to design their embedded systems to last more than a few years. Yes, a Blu-ray player is an embedded system that is probably running Linux.

As embedded designers, we need to create more things to last. In many cases, the software we write and the hardware on which it runs still need to be running 10, 20, even 30 years from now. Of course we know that deep-space systems need that kind of lifetime. But is that true for the systems we design? As more low-tech devices become "connected," consumers are not going to want to swap out their toilet or furnace every three to four years. My company is currently working on a smart electric meter. When was the last time you swapped out your electric meter? They are supposed to last 30 years.

How can you make things last with something as complex as a Linux flash memory system? This article explains how to do that. Although many things contribute to a robust system, I will only discuss making the flash memory system robust. That is why this column is called "Embedded in Thin Slices."

THE PROBLEM: DATA RETENTION

In the first article in this series, I mentioned that most NAND flash devices are specified to only retain data for 10 years. A 4-Gb SPI flash we use specifies 20 years for data retention. What this means is that, independent of the wear-leveling issues I discussed in Part 1, if you just write the data once to these devices, you can only count on them to keep that data for 10 or 20 years, respectively.

For the bulk of our systems, the file system uses wear leveling (even on read-only data), which greatly extends the data retention life of these devices. It does this by constantly moving data around the flash memory. However, in one of our designs that should last more than 20 years, there are three other regions in our NAND flash memory (called partitions) that are not part of the file system and are usually never changed for the products' life. They are the bootstrap loader (u-boot), the u-boot environment variables, and the Linux kernel. Thus the data retention problem is real for these partitions. In 10 years, these regions will start dying.

Some flash memory characteristics can cause other errors that can prevent us from creating a robust 24/7 long-life embedded system. For example, both NAND and NOR flash memory can exhibit an error called bit flipping. This happens when a single bit is reported as reversed or is actually reversed from the required state. I have already talked about how flash memory has limited write/erase cycles. In addition, flash memory can wear out due to too many reads! I didn't know that until I dug deep into a Micron Technology Technical Note (see Resources). In most systems, this is not an issue because flash data is usually seldom read. In the rare case where you are designing a Linux system that executes instructions directly out of the flash memory or one that repetitively reads data directly from the flash memory, you should be aware of the fact that flash memory wears out on reads as well as writes. Finally, all flash memory devices now come from the factory with bad blocks. The manufacturers guarantee that the first block is not bad, but after that, it is up to your flash memory manager to handle these bad blocks.

SOLUTIONS

Fortunately, there are workable solutions to each of these issues to extend a flash memory system's life. Applying error-correcting code (ECC) is the first line of defense and is widely used with flash devices. In addition, a general redundancy is relatively inexpensive to implement in hardware and software if you plan for it up front. Let's look at each of these and see what we can learn.

ERROR-CORRECTING CODE

ECC is used to detect and correct errors. One-bit ECC can detect and correct any single-bit errors. This is useful for correcting bit-flipping errors. Four-bit ECC can detect and correct up to 4 bits of error in a single page (typically 512 bytes). Both techniques accomplish this by storing redundant data on the page that enables it to restore the data to its original value if a bit failure should occur. Flash memory devices come with extra space on each page to store ECC data.

Our company designed a major system for a customer and missed the fact that the flash chip required 4-bit ECC while the Linux kernel and u-boot for the chip we were using only supported one-bit ECC. After about two years and 20,000 units were deployed, systems started failing. This was a costly error that affected a large percentage (relatively) of the fleet before we were able to remotely update the software. Once 4-bit ECC was implemented, the failure rate went basically to zero in our test lab.

We hope we see the same in the field. At the time our product was released, neither the Linux kernel nor the boot loader supported four-bit ECC. Thankfully, by the time the problem surfaced, these features were supported in the kernel and in u-boot. Thus we did not have to create these algorithms and write this code.

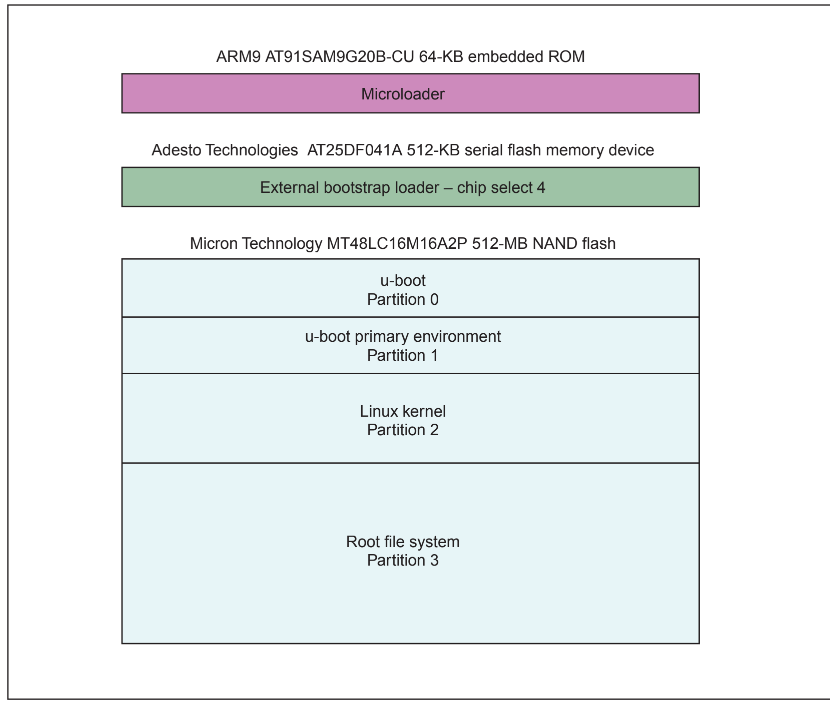


FIGURE 1
This is the memory layout of memory regions required to bring up one of my company's embedded Linux designs with no redundancy.

COLUMNS

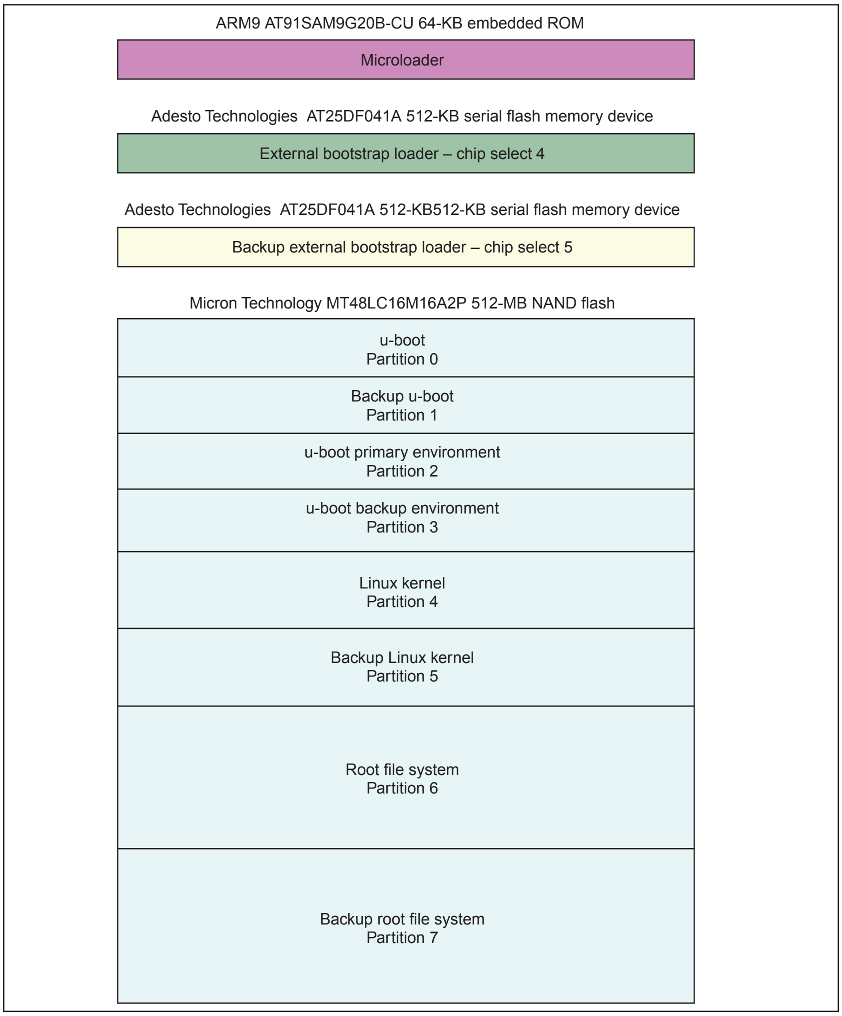


FIGURE 2
This memory layout creates the most reliable design by providing redundant memory regions for the external bootstrap, u-boot, u-boot's environment, the kernel, and the file system.

The moral of the story: Thoroughly read all datasheets and their associated errata for every device in your system. Know what your OS does underneath you. Even if Linux says: "We'll take care of it for you." Don't trust them! Tests run at the Jet Propulsion Laboratory showed that without ECC, 1-bit errors started occurring at one fifth the life of a particular memory device.

ECC can be supported in either software or hardware if your processor and the OS support this feature. The ARM9 processor we used on the previously mentioned product only supports 1-bit ECC and is not useful with the memory chips we use.

REDUNDANT FLASH REGIONS

Before I describe a typical Linux system with redundancy that we design, let me describe how the system would start if it *didn't* have redundant flash regions. This description applies to an ARM9 architecture.

Figure 1 shows different types of memory and the roles they play at startup. The purple memory is ROM stored in the ARM9. The green memory is SPI flash memory stored in an external chip. The blue memory is the NAND flash memory stored in one or more chips.

At power-up, the system runs a microloader stored in ROM in the ARM9. This microloader can load a bootstrap loader into internal SRAM from any of several devices. Our system loads the bootstrap loader from a 4-Gb SPI flash on chip select 4. Once loaded, the microloader starts the bootstrap loader running out of internal static RAM (SRAM). The bootstrap loader loads our Linux boot loader (u-boot) from NAND flash Partition 0 into dynamic RAM (DRAM). Once complete, it transfers control to u-boot, which sets up the hardware as specified in the configuration environment block (stored in Partition 1 of the NAND flash). It then loads the Linux kernel from another NAND flash Partition (2) into DRAM. Control is then transferred to the Linux kernel, which mounts the file system in Partition 3 and then starts the various threads from applications stored on the file system.

There are two ways this can get you in trouble. Since the data retention lifetime is only 10 years for the NAND flash, Partitions 0–2 never get "wear leveled" and thus will start "failing" after 10 years. The life of these partitions can simply be extended by rewriting the partition. Squirreling away copies of these partitions (compressed or not) can enable the system to occasionally (once per year) rewrite them, which will dramatically extend the life of these regions. In the end, you end up with something like what is shown in **Figure 2**. Here the microloader attempts to load the

circuitcellar.com/ccmaterials

RESOURCES

Y. Chen "Flash Memory Reliability: NEPP 2008 Task Final Report," NASA Jet Propulsion Laboratory, 2008.

B. Japenga, "Embedded File Systems (Part 1): Linux File Systems," *Circuit Cellar* 279, 2013.

———, "Embedded File Systems (Part 2): File System Integrity," *Circuit Cellar* 281, 2013.

Micron Technology, Inc., "NAND Flash Design and Use Considerations Introduction," Technical Note TN-29-17, Rev B 2010.

SOURCES

AT25DF041A 512-KB Serial interface flash memory device
Adesto Technologies Corp. | www.adestotech.com

ARM9 processor
ARM, Ltd. | www.arm.com

MT48LC16M16A2P Single data rate synchronous dynamic RAM
Micron Technology, Inc. | www.micron.com

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.

bootstrap loader from chip select 4. If it fails to load, it will load it from another SPI device (yellow) on chip select 5. The loader, running in internal RAM, will attempt to load and run u-boot from either Partition 0 or Partition 1. Once running, u-boot will attempt to load the configuration from either Partition 2 or Partition 3. Finally, u-boot will attempt to load the kernel from either Partition 4 or Partition 5. All of these features are built into the open-source software for these packages.

Additional logic, which is not built into Linux, requires you to check the file system's integrity at start up to a previously defined "manifest" of CRC or MD5 sums for each file. Should there be any issues, the shadow file system partition (Partition 7) is mounted and the damaged files corrected.

Building the logic into each of the loaders to check data integrity and loading the alternate partition can enable the design to be even more robust than before. If one of the partitions is corrupted for any reason, the loader will choose the backup partition (or device) to boot. Once loaded, Linux can be flagged to rewrite the bad region. Out of the box, Linux even provides a means to write the bootstrap loader with a file image. This scheme also provides power-down protection for power outages that occur when the partition is being written (since it is not being used).

Since flash memory devices can cascade failures (i.e., failures in one page can cause failures in another page), the embedded designer needs to keep that in mind when designing a robust system. Sometimes this can be solved by keeping your redundant data on a separate memory device rather than just a separate partition. Without knowing the physical layout of the chip, the designer doesn't know which partitions are physically next to each other.

ERROR-CORRECTING CODE AND REDUNDANCY

Some embedded systems are throwaways. We have designed some systems that are only run once. Other systems must last for more than 10 years. Flash memory devices are becoming more dense and less perfect. Not long ago, memory device manufacturers would not ship bad blocks in their devices. Now the devices routinely have many bad blocks. As embedded systems designers, we need to be aware how our flash memory systems work, what the failure modes are, and how they need to be reliably interfaced. Through redundancy and ECC you have added two tricks to your playbook. But just a thin slice!

The next article in this series will describe two alternate file systems available under Linux: a compressed ROM file system (cramfs) and a RAM file system.

Wireless Data Links (Part 1)

Radio Communications

Wireless data-carrying devices are ubiquitous. This article series examines radio communications principles and provides a helpful overview of wireless communications.

By George Novacek (Canada)

Over the last two decades, wireless data communication devices have been entering the realm of embedded control. The technology to produce reasonably priced, reliable, wireless data links is now available off the shelf and no longer requires specialized knowledge, experience, and exotic, expensive test equipment. Nevertheless, to use wireless devices effectively, an engineer should understand the principles involved.

The invention of radio marked the beginning of the electronics engineering discipline as we know it today. In the Western world, we tend to credit Guglielmo Marconi with achieving the first long-distance radio communications. The Russians claim Alexander Stepanovich Popov to be the sole inventor of radio and celebrate May 7th as the Day of Radio. However, the long-distance radio communications almost simultaneously demonstrated by Marconi and Popov in the mid-1890s would have been impossible without the pioneering work of scientists throughout the 19th century, many of whom are recognized as the giants of science. Among them are Hans Christian Ørsted, André-Marie Ampère, Joseph Henry, Michael Faraday, Heinrich Rudolph Hertz, James Clerk Maxwell, and others.

WIRELESS DEVICES

This article will address low-power, data-carrying wireless links that can be used in control systems. Even with this limitation, it is a vast subject, the surface of which can merely be scratched. Today, we can purchase ready-made, low-power, reliable radio interface modules with excellent performance for an incredibly low price. These devices were originally developed for noncritical applications (e.g., garage door openers, security systems, keyless entry, etc.). Now they are making inroads into control systems, mostly for remote sensing and computer network data exchange. Wireless devices are already present in safety-related systems (e.g., remote tire pressure monitoring), to say nothing about their bigger and older siblings in remote control of space and military unmanned aerial vehicles (UAVs).

In the US, low-power radio devices operate under the Federal Communications Commission (FCC) Part 15 rule, which means they do not require individual licensing. Other countries have similar regulations. The transmitters emit low power, usually on the order of milliwats, most often operating in 300-to-900-MHz and 2.4-GHz frequency bands. In addition, 315-, 433.9-, and 900-MHz and 2.4-GHz bands are popular. Inexpensive modules operating on those frequencies have typical open field range of about 300′ (100 m).

WIRELESS COMMUNICATIONS

Let's review some fundamental wireless communications principles. RF circuits' essential building blocks are resonators, such as the inductor/capacitor (LC) circuits shown in **Figure 1**. In many present-day devices the LC tank circuits were replaced by ceramic or surface acoustic wave (SAW) resonators. But for some functions (e.g., impedance matching), LC networks are hard to replace.

Resonant frequency of the tuned circuits in **Figure 1** is calculated:

$$f_R = \frac{1}{2 \times \pi \times \sqrt{L \times C}} \text{ Hz}$$

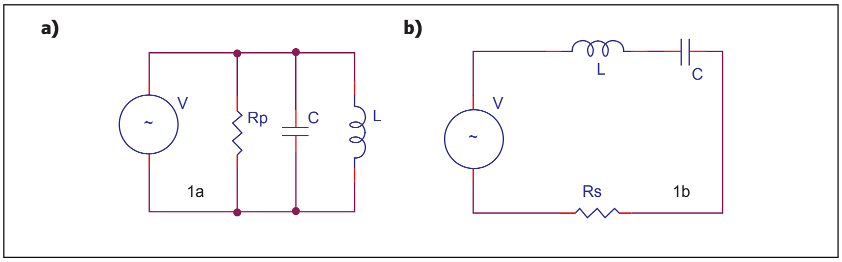


FIGURE 1
a—A parallel resonant circuit is shown. **b—**This is a series resonant circuit.

Assuming the capacitor C = 100 pF and the inductor is 1 µH, the resonant frequency f_R = 15.92 MHz. In a perfect world, with ideal components, the parallel resonant circuit's impedance would be infinite (i.e., an open circuit). All the current from the source would only flow through the resistor Rp. Conversely, the series resonant circuit's impedance would be zero (i.e., a short circuit) and the current from the source would be determined only by the resistor Rs. In both cases, the red trace in **Figure 2** would become a vertical line at the resonance.

Unfortunately, the world is not perfect. A real resonant circuit's frequency response looks more like the red trace in **Figure 2**. Its shape is determined by the circuit's quality factor Q, which is a nondimensional value. Its magnitude depends on the value of the resistance Rp or Rs. Rp and Rs represent a composite of the inductor's ohmic resistance, the capacitor leakage, external loads, and other parasitic losses. An actual resistor may be intentionally added to lower the Q to increase the bandwidth (BW). The Q of a parallel resonant circuit can be calculated:

$$Q = Rp \times \sqrt{\frac{C}{L}} = \frac{Rp}{2 \times \pi \times f_R \times L} = 50$$

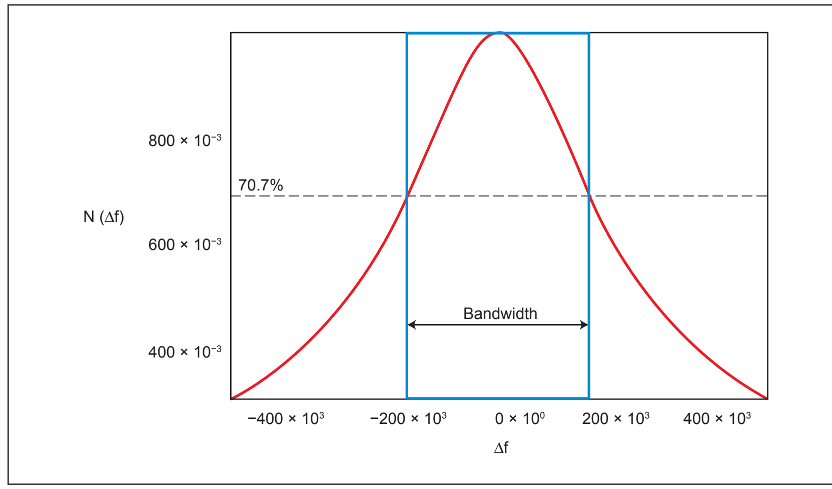


FIGURE 2
This is the frequency response and bandwidth of a parallel resonant LC circuit. A series circuit graph would be inverted.

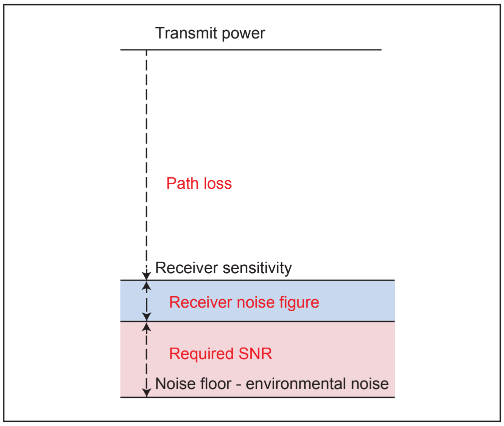
for the LC components given above and Rp = 5 kΩ. Similarly, the Q of a series circuit can be calculated:

$$Q = \frac{1}{Rs} \times \sqrt{\frac{L}{C}} = \frac{2 \times \pi \times f_R \times L}{Rs} = 50$$

when Rs = 2Ω. The reciprocal of Q is called ξ (Greek letter "Xi"). It is referred to as damping and for Q = 50 it equals 2×10^{-2}.

The bandwidth is the frequency deviation

FIGURE 3
The relationship between noise and receiver sensitivity is shown.



from the resonance where the magnitude of the voltage across the parallel or current in the series resonant circuit drops by 3 dB (i.e., to 70.7% of the magnitude at resonance). It is calculated:

$$BW = \frac{f_R}{Q} = 318.3 \text{ kHz}$$

for the components in this example. This is also shown in **Figure 2**. The frequency response, the red trace in **Figure 2**, can be calculated and plotted by:

$$N = \frac{1}{\sqrt{1 + \left(\frac{2 \times Q \times \Delta f}{f_R}\right)^2}}$$

where N is the normalized gain on the ordinate (N = 1 in resonance), Q is the quality factor, Δf is frequency deviation from the resonance, and f_R is the resonant frequency.

BANDWIDTH

The bandwidth affects receiver selectivity and/or a transmitter output spectral purity. The selectivity is the ability of a radio receiver to reject all but the desired signal. Narrowing the bandwidth makes it possible to place more transmitters within the available frequency band. It also lowers the received noise level and increases the selectivity due to its higher Q. On the other hand, transmission of every signal but a nonmodulated, pure sinusoid carrier—which, therefore, contains no information—requires a certain minimum bandwidth. The required BW is determined by the type of modulation and the maximum modulating frequency.

For example, AM radios carry maximum 5-kHz audio and, consequently, need 10-kHz bandwidth to accommodate the carrier with its two 5-kHz sidebands. Therefore, AM broadcast stations have to be spaced a minimum of 20 kHz apart. However, narrowing the bandwidth will lead to the loss of parts of the transmitted information. In a data carrying systems, it will cause a gradual increase of the bit error rate (BER) until the data becomes useless. At that point, the bandwidth must be increased or the baud rate must be decreased to maintain reliable communications.

An ideal bandwidth would have a shape of a rectangle, as shown in **Figure 2** by the blue trace. Achieving this to a high degree with LC circuits can get quite complicated, but ceramic resonators used in modern receivers can deliver excellent, near ideal results.

SENSITIVITY AND NOISE

Another characteristic of a radio receiver is its sensitivity. Receiver manufacturers sometimes publish fantastic sensitivity numbers in a sub-microvolt range. Those numbers are meaningless if they are not accompanied by other important characteristics, such as the signal-to-noise ratio (SNR), bandwidth, intermodulation (IM), distortion and so forth. Today, it is more common to express sensitivity in dBm, which is the received signal level in decibels (dB) required for 1-mW output. But this would also be meaningless if not accompanied by the SNR, IM, and BW—or, more appropriate for wireless data links, by BERs at a certain baud rate. Good data receivers feature sensitivity around –90 to –105 dBm at 10^{-2} BER. The maximum achievable baud rate depends on the type of carrier modulation and the bandwidth.

And then there is the ever-present noise. Random noise is inherent to every electronic system. All passive components produce it, active components produce it, and we are surrounded by environmental noise generated by the universe, industries, appliances, and so forth. Noise average value is expressed as:

$$\overline{i(t)} = \lim_{T \to \infty} \frac{1}{T} \times \int_0^T i\,(t) \times dt = 0$$

In other words, over time, the noise averages to zero. The frequency spectrum of noise is continuous (i.e., it contains all frequencies in the band). Its current spectral

RESOURCES

US Department of Commerce, "United States Frequency Allocations: The Radio Spectrum," 2003.

Government of Canada, "Radio Spectrum Allocation in Canada," 2001.

circuitcellar.com/ccmaterials

ABOUT THE AUTHOR

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

density per 1 Hz is:

$$\omega_I(f) = \frac{di^2}{dt}$$

If this density remains constant over a wide frequency range, the noise is called white noise.

In practical terms, it means that we cannot keep improving receiver sensitivity and thus the communications range by increasing the receiver's electrical gain forever. **Figure 3** shows how receiver sensitivity is affected by noise.

The receiver input signal needs a certain SNR to be correctly decoded. It can be as low as 3 dB depending on many factors. At some point, its electrical gain, its internally generated noise, and the required SNR reach the noise floor and no more improvement of sensitivity is possible. Even today's inexpensive, off-the-shelf receiver modules claim −92 dBm and better sensitivity (other characteristics rarely provided). This is often below the environmental noise level. With no RF signal present, the output chatters due to the noise. Once a signal with sufficient SNR is detected, the automatic gain control (AGC) squelches the noise.

When the noise is added to a valid signal, the combination is no longer random and statistical techniques can be used to distinguish between the two. Correlation techniques have long existed to recover signals buried deep in the noise. However, correlation requires a lot of computing power, so we'll have to wait a while before it becomes available in low-power, low-cost RF links.

In Part 2 of this article series I'll discuss transmitters and antennas.

THE DARKER SIDE

Dielectric Absorption

Dielectric absorption occurs when a capacitor that has been charged for a long time doesn't completely release its voltage during a quick discharge. This article uses experiments to describe the effects of dielectric absorption.

By Robert Lacoste (France)

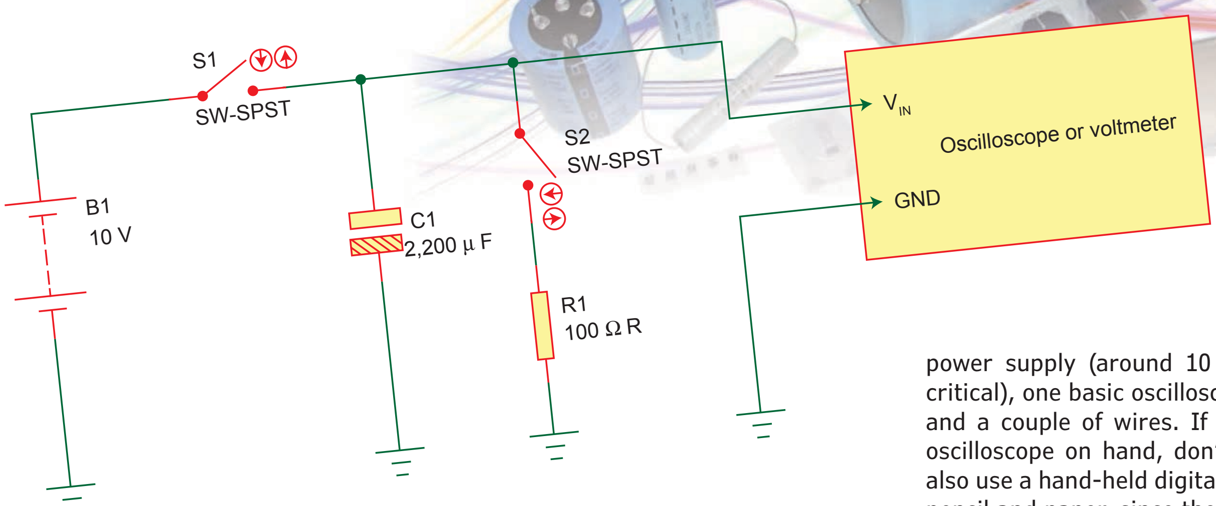


FIGURE 1
The setup for experimenting with dielectric absorption doesn't require more than a capacitor, a resistor, some wires and switches, and a voltage measuring instrument.

Electronic parts can occasionally be challenging to work with. This can happen with even basic components, such as capacitors. I have already shown you that capacitors can behave as inductors when they are inadvertently used above their resonant frequency (see my article "Parasitic Components: When Capacitors Behave Like Inductors," *Circuit Cellar* 245, 2010). This month, I will describe another strange behavior of capacitors: dielectric absorption. This behavior is amusing, as this surprising phenomenon is easy to demonstrate with basic test instruments. So read on, enjoy, and more importantly don't hesitate to test it yourself!

A BASIC EXPERIMENT

Instead of starting with theory, I'll begin with a simple experiment. Go down to your cellar, or your electronic playing area, and find the following: one large electrolytic capacitor (e.g., 2,200 µF or anything close, the less expensive the better), one low-value discharge resistor (100 Ω or so), one DC power supply (around 10 V, but this is not critical), one basic oscilloscope, two switches, and a couple of wires. If you don't have an oscilloscope on hand, don't panic, you could also use a hand-held digital multimeter with a pencil and paper, since the phenomenon I am showing is quite slow. The only requirement is that your multimeter must have a high-input impedance (1 MΩ would be minimum, 10 MΩ is better).

Figure 1 shows the setup. Connect the oscilloscope (or multimeter) to the capacitor. Connect the power supply to the capacitor through the first switch (S1) and then connect the discharge resistor to the capacitor through the second switch (S2). Both switches should be initially open. **Photo 1** shows you my simple test configuration.

Now turn on S1. The voltage across the capacitor quickly reaches the power supply voltage. There is nothing fancy here. Start the oscilloscope's voltage recording using a slow time base of 10 s or so. If you are using a multimeter, use a pen and paper to note the measured voltage. Then, after 10 s, disconnect the power supply by opening S1. The voltage across the capacitor should stay roughly constant as the capacitor is loaded and the losses are reasonably low.

Now switch on S2 long enough to fully discharge the capacitor through the 100-Ω

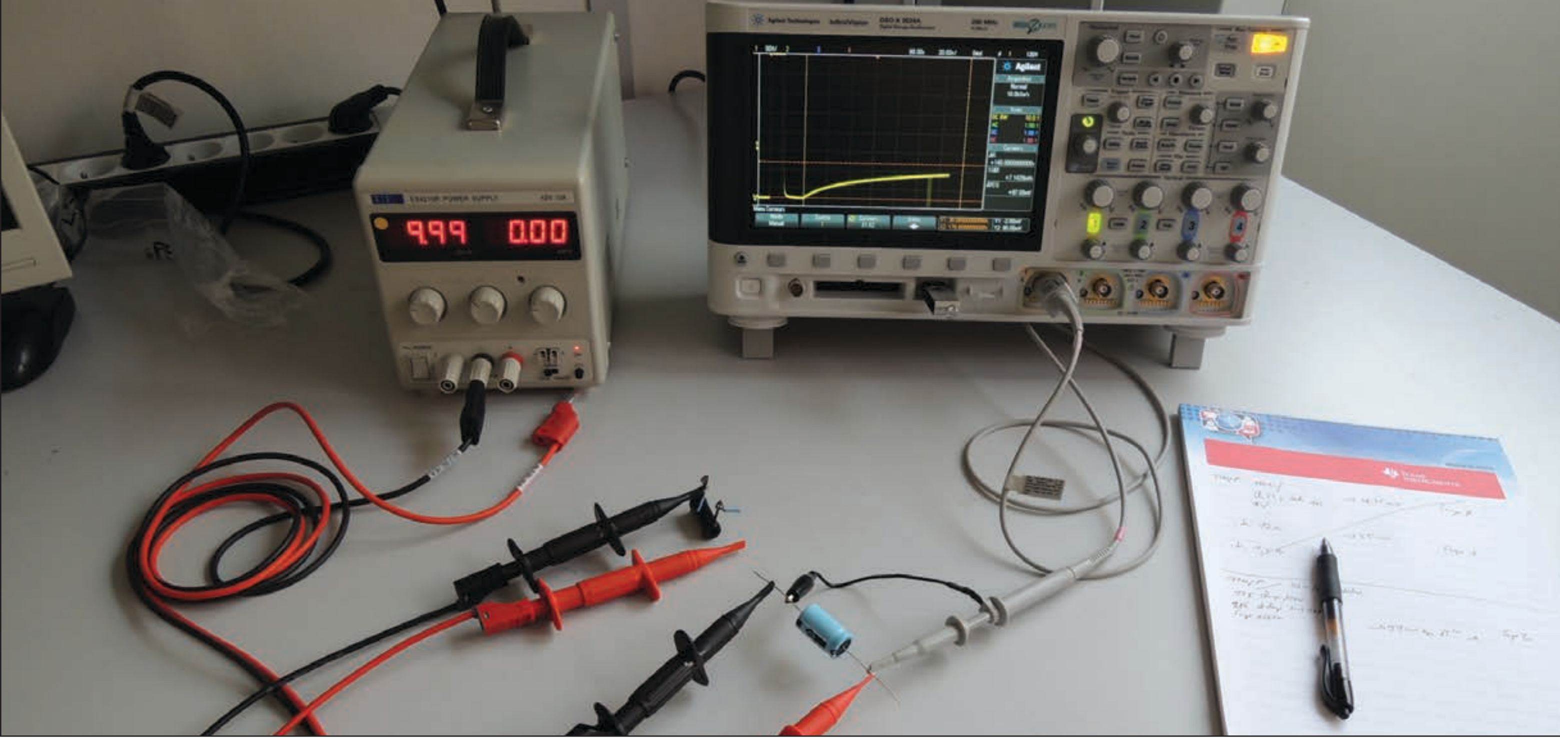


PHOTO 1
My test bench includes an Agilent Technologies DSO-X-3024A oscilloscope, which is oversized for such an experiment.

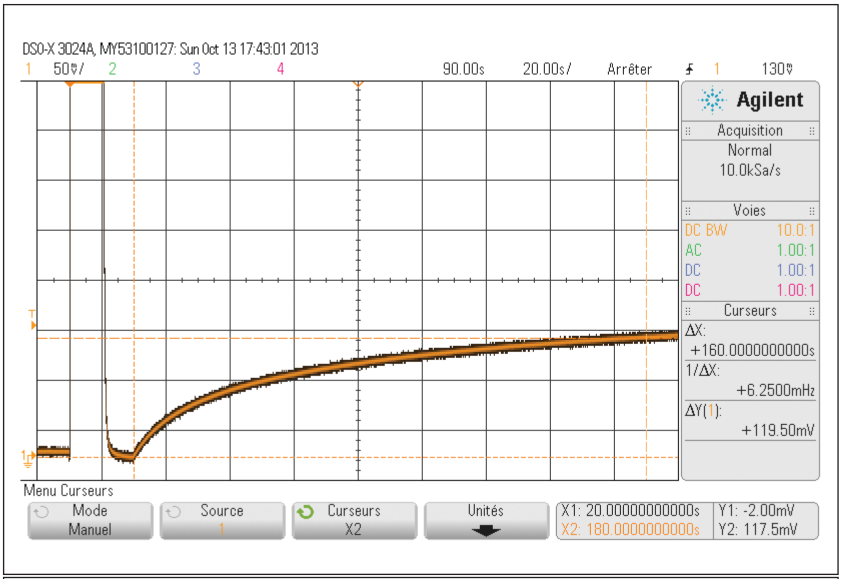


PHOTO 2
I used a 2,200-µF capacitor, a 100-Ω discharge resistor, and a 10-s discharge duration to obtain this oscilloscope plot. After 2 min the voltage reached 119 mV due to the dielectric absorption effect.

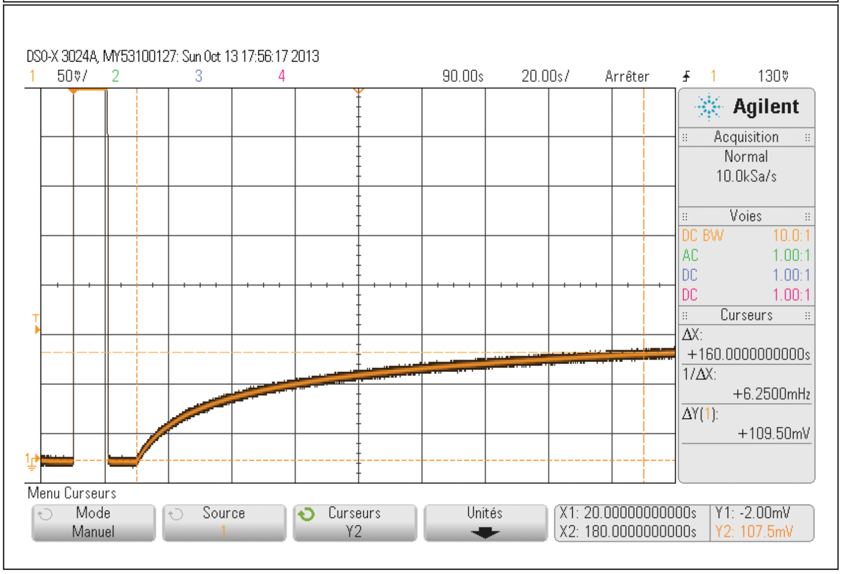


PHOTO 3
Replacing the 100-Ω discharge resistor with a 1.5-Ω resistor doesn't significantly change the plot as compared with Photo 2.

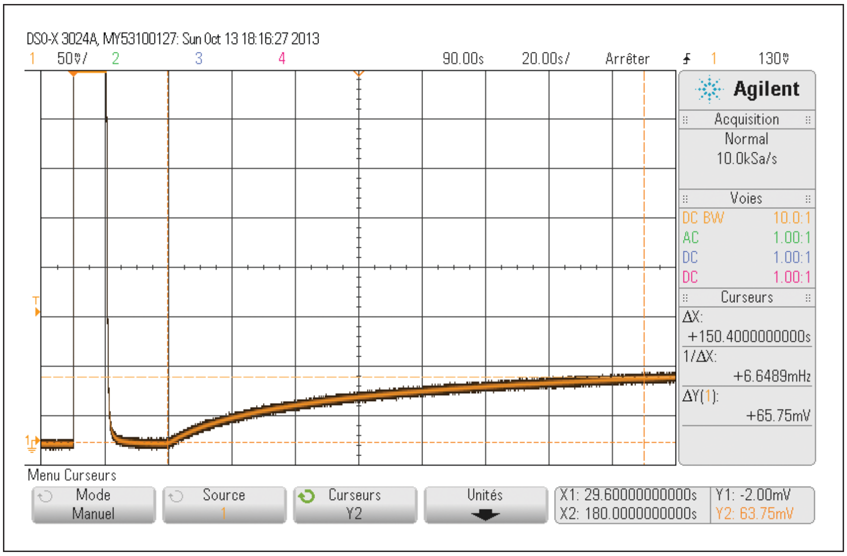


PHOTO 4
I used a 100-Ω resistor with the discharge duration extended to 20 s to record this plot. The final voltage is roughly divided by two compared with the initial setup.

resistor. As a result of the discharge, the voltage across the capacitor's terminals will quickly become very low. The required duration for a full discharge is a function of the capacitor and resistor values, but with the proposed values of 2,200 µF and 100 Ω, the calculation shows that it will be lower than 1 mV after 2 s. If you leave S2 closed for 10 s, you will ensure the capacitor is fully discharged, right?

Now the fun part. After those 10 s, switch off S2, open your eyes, and wait. The capacitor is now open circuited, at least if the voltmeter or oscilloscope input current can be neglected, so the capacitor voltage should stay close to zero. But you will soon discover that this voltage slowly increases over time with an exponential shape.

Photo 2 shows the plot I got using my Agilent Technologies DSO-X 3024A digital oscilloscope. With the capacitor I used, the voltage went up to about 120 mV in 2 min, as if the capacitor was reloaded through another voltage source. What is going on here? There aren't any aliens involved. You have just discovered a phenomenon called dielectric absorption!

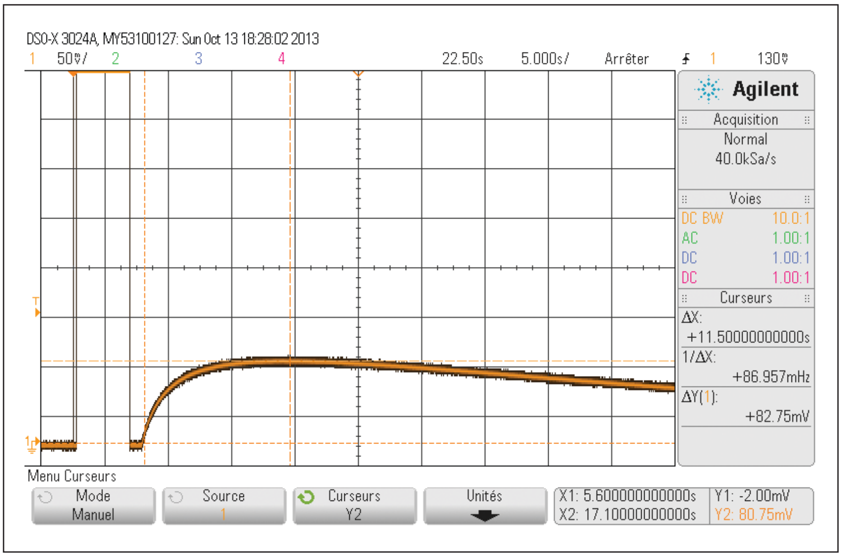


PHOTO 5
With a 4.7-µF electrolytic capacitor and a 2-s discharge through a 100-Ω resistor, the voltage across the capacitor rises to 83 mV in 12 s then slowly decreases due to losses.

EXPERIMENTAL VARIANTS

Before explaining the root cause of this effect, it is useful to try some other experiments. You may think that the resistor value was too high for a full discharge. I did the test for you and replaced the 100-Ω resistor with a 1.5 Ω resistor I had on hand.

Photo 3 shows the result with the same 10-s discharge duration. As expected, the discharge is much faster due to the lower-value discharge resistor. But, after 2 min, the voltage across the capacitor is 110 mV, which is very close to the 120 mV I got in the first test. So the discharge resistor's value doesn't seem to have a great influence.

I replaced the 100-Ω resistor and instead of reducing its value, I increased the discharge phase's duration. Interestingly, if S2 was kept closed for 20 s rather than 10 s, then the voltage across the capacitor after 2 min would then be 65 mV, which is roughly half the voltage I got in the initial test (see **Photo 4**). It looks like there is a close link between the discharge duration and this phantom voltage.

The last interesting test was to check other electrolytic and tantalum capacitors. Of course the time scale and/or resistor value must be adapted if the capacitor is significantly smaller than 2,200 µF, but I always found the same behavior. Voltage and time values could change, but the overall shape stayed similar.

Photo 5 shows an example with a low equivalent series resistance (ESR) 4.7-µF/63-V electrolytic capacitor. After a 5-s charge and a 2-s discharge over 100 Ω, the voltage across the capacitor rose up to 83 mV in 12 s then slowly decreased. This last phase was due to this capacitor's internal discharge combined with the oscilloscope's small input current.

Then I replaced the 4.7-µF electrolytic capacitor with a 4.7-µF ceramic capacitor. The phantom voltage increase was drastically lower, close to 20 mV even if this measurement was more difficult to do with the oscilloscope and its 1:10 probe.

Let me summarize what these experiments show. With certain types of capacitors—mainly electrolytic and tantalum—the capacitor doesn't seem fully discharged after a fast discharge cycle. More specifically, the capacitor seems to autonomously recharge itself but with a slow time scale (several seconds or minutes). The final voltage depends on the capacitor, the initial voltage, and the discharge duration but not so much on the discharge resistor value. As an order of magnitude, in my tests this final voltage was close to 1% of the initial charge voltage.

EXPLANATIONS

To understand where this dielectric absorption comes from, I need to move from electronics to chemistry. You probably know that a capacitor is built using two parallel conducting planes, between which an electric field is applied through the external voltage. Between these two planes there is usually a dielectric material ensuring insulation. This material is the cause of our troubles.

According to *Wikipedia*, this dielectric, in particular in the case of electrolytic capacitors, is composed of molecules "glued" between each other through chemical links.[1] The vast majority of the molecules present an electrical polarization, meaning that their electrons are located more on one end of the molecule than the other. When an electric field is applied between the capacitor's plates, the dielectric

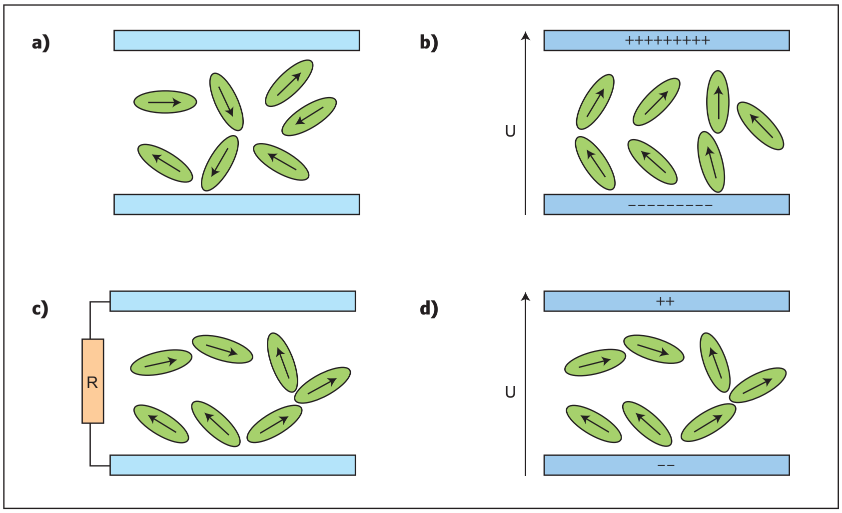


FIGURE 2
a—The dielectric molecules are randomly oriented. **b—**When an electric field is created between the capacitor's planes the molecules orient themselves to the field direction. **c—**Discharging the capacitor frees the orientation of the molecules, but this takes time. **d—**If the load is opened too soon, the molecules are still oriented. This orientation propagates to the planes and recreates a voltage across the capacitor's terminals.

COLUMNS

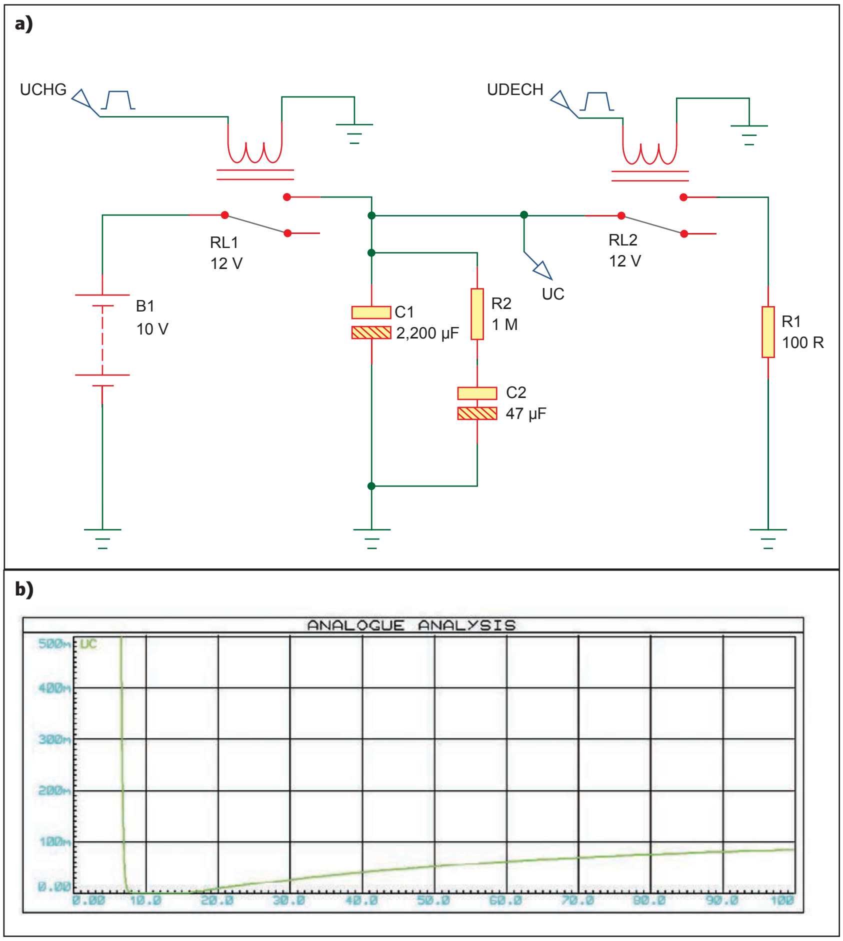


FIGURE 3

a—The dielectric absorption can be modeled as a small capacitor in series with a large resistor and connected across the main capacitor. **b—**This small simulation using Labcenter Electronics's Proteus VSM design suite shows that the behavior can effectively be modeled very close to the experiment.

molecules tend to align with this electric field. However, this move is slowed by the links between these molecules and their neighbors. This has two consequences. First, there is some energy dissipated in the dielectric, which results in the so-called dielectric loss of the capacitor. Second, this alignment will take some time (see **Figure 2**).

When the capacitor is discharged, the electric field is nullified so the molecules will relax and return to their non-oriented state, but this takes some time. If the capacitor is discharged quickly enough, a significant part of the molecules will still be polarized at the end of the discharge and the polarization will slowly propagate to the capacitor's plates. This is what we have just experimentally demonstrated!

This explains why the measured recharge depends on the discharge's duration and the kind of dielectric. By the way, an air capacitor doesn't suffer from any significant dielectric absorption and a ceramic one is far less impacted than an electrolytic variant.

Bob Pease's *Electronic Design* article "What's All This Soakage Stuff Anyhow?" provides a complete analysis of this phenomenon. In particular, Pease reminds us that the model for a capacitor with dielectric absorption effect is a big capacitor in parallel with several small capacitors in series with various large resistors.

I used Labcenter Electronics's Proteus VSM design suite to build a small simulation. The result is very close to the experiment, as demonstrated on **Figure 3**.

Pease, who passed away in 2011, worked for National Semiconductors (now Texas Instruments). He left behind a wonderful set of publications about analog circuits. Texas Instruments republished all his articles on its website (see Resources). I strongly encourage you to visit the site and read (and reread) these articles.

Let's return to dielectric absorption. The subject is important enough that there is a standardized method with which to measure it. It is specified in the US Department of Defense's military standard MIL-PRF-19978 which is available online (see Resources). Basically this is the same method I used in my examples, but the values differ. In this standard, the capacitor must first be charged for 1 h at a rated DC voltage then discharged for 10 s through a 5-Ω resistor. The regained voltage is measured during the following 15 min with a very high impedance millivoltmeter, specifically above 10,000 MΩ. The capacitor's dielectric absorption is the ratio between the regained voltage and the initial voltage, expressed as a percentage.

Capacitor manufacturers publish tables

circuitcellar.com/ccmaterials

REFERENCE

[1] *Wikipedia*, "Dielectric Absorption."

RESOURCES

R. Lacoste, "Parasitic Components: When Capacitors Behave Like Inductors," *Circuit Cellar* 245, 2010.

B. Pease, "What's All This Soakage Stuff Anyhow?" *Electronic Design*, 1998.

Texas Instruments, Inc. (formerly National Instruments, Corp.), "The Best of Bob Pease," 2012.

US Department of Defense, "Performance Specification: Capacitors, Fixed, Plastic (or Paper-Plastic) Dielectric (Hermetically Sealed on Metal, Ceramic or Glass Cases), Established and Non-Established Reliability General Specification," MIL-PRF-19978, 2009.

SOURCES

DSO-X 3024A Digital oscilloscope
Agilent Technologies, Inc. | www.agilent.com

Proteus VSM design suite
Labcenter Electronics | www.labcenter.com

ABOUT THE AUTHOR

Robert Lacoste lives in France, near Paris. He has 24 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 if you don't forget to put "darker side" in the subject line to bypass spam filters.

showing the dielectric absorption coefficients of several technologies. **Table 1**, which was reproduced from *Wikipedia*, shows you some typical values.[1]

WRAPPING UP

Imagine you have to do some maintenance work on a cathode ray tube (CRT)-based TV or any other high-voltage device. You disconnect it from the power line, open it, and of course you know that you need to discharge all high-voltage capacitors before putting your hands inside. So you take a discharge resistor, apply it on that large 10-kV/1-µF capacitor for 5 or 10 s to be safe, disconnect it, and place your finger there. If you are not lucky, the voltage could have already increased to some hundreds of volts thanks to dielectric absorption, and this could be lethal. Even smaller 400-V capacitors may recover a voltage high enough to be dangerous to humans or, more commonly, to cause damage to other electronic parts nearby.

Dielectric absorption could also be painful if you have to design a high-precision sample-and-hold circuit. In such a circuit, the input voltage is quickly connected to a storage capacitor, which is then open circuited. But due to dielectric absorption phenomenon, the capacitor will then inevitably recover a part of its previous voltage. This may create erroneous results. What's worse, these results will depend on the past input voltages. For these applications, specific low dielectric absorption capacitors are a must.

In the past, polystyrene, polypropylene, or even Teflon technologies were often used for these applications. Today, a sample-and-hold circuit is usually bought as an integrated chip with its on-chip capacitor specifically designed for this application.

Dielectric absorption can also introduce frequency-dependent distortion in audio circuits, which is why audiophiles spend a lot on exotic capacitor technologies. (Again, refer to Pease's article for some discussions on that topic.)

Here we are. I hope this article encouraged you to try dielectric absorption. Do so and do it now. It is fun and very easy!

Type of Capacitor	Dielectric Absorption
Air and vacuum capacitors	Not measurable
Class-1 ceramic capacitors, NP0	0.6%
Class-2 ceramic capacitors, X7R	2.5%
Polypropylene film capacitors (PP)	0.05% to 0.1%
Polyester file capacitors (PET)	0.2% to 0.5%
Polyphenylene sulfide film capacitors (PPS)	0.05% to 0.1%
Polyethylene naphthalate film capacitors (PEN)	1% to 1.2%
Tantalum electrolytic capacitors with solid electrolyte	2% to 3%, 10%
Aluminum electrolytic capacitors with non-solid electrolyte	10% to 15%

TABLE 1
This table provides the common dielectric absorption factors of classic capacitors. For electrolytics, the self recharge can be up to 10% to 15% of the initial voltage. (Information courtesy of *Wikipedia)*

FROM THE BENCH

PHOTO 1
My parking ticket, train ticket, subway pass, and event pass each used a different data technology. Some day we may get it all together. E-ZPass anyone?

A Visit to the World Maker Faire

In September 2013, Jeff traveled to New York City to attend the World Maker Faire, an annual convention for hackers, makers, and DIYers. In this article, Jeff shares some interesting maker projects and unique 3-D applications and provides some insight into the future of technology.

Jeff Bachiochi (US)

The car gently rocked to the clickity-clack of wheels rolling over rail joints. The image of my world was limited to mysterious silhouettes that flit past the window. It was easy to daydream at night due to the lack of visual stimulation. The lonely parking lot I had just left was devoid of life except for the lone machine that offered up a receipt when I fed it the appropriate currency. I had printed my round-trip Amtrak ticket at home. Its 2-D QR code image contained my reservation information. The conductor's hand-held reader confirmed my legal authorization to ride train #143 for the next 2 h and 42 min.

When dawn began to breathe life into the day, most passengers were unaware of the change as the open Wi-Fi had already taken them hostage. And then, in a flash, it was black. We were descending into a massive tangle of railways, subways, sewers, and utilities all built beneath the foundations of a thousand buildings and streets that make up the "Big Apple." From the railway at New York City's Penn Station, you can make subway connections without ever seeing daylight. A swipe of my MetroCard gave access to the 7 train and an hour-long ride to the 111th street station. From there I joined a pilgrimage en route to the New York Hall of Science, which was host to the fourth annual World Maker Faire New York. At the entry gate, my 2-D data matrix image was scanned on my Maker Faire ticket (see **Photo 1**). Ahead were acres of makers, who served up a fun and educational experience.

WORLD MAKER FAIRE

The World Maker Faire is part science fair and part country fair. Makers are DIYers. The maker movement empowers everyone to build, repair, remake, hack, and adapt all things. The Maker Faire shares the experiences of makers who have been involved in this important process, which for so long had been lost or repressed. Social media keeps us in constant contact and can educate (e.g., through Google and YouTube videos), but it can't replace the feeling you can get from hands-on live interaction with people and the things they have created.

It should be noted that not all Maker Faire exhibitors are directly involved with technology. Some non-technological projects on display included the "Art Car" from Pittsburgh, which is an annual revival of an old clunker turned into a drivable art show on wheels. There was also the life-size "Mouse Trap" game, which was quite the contraption and just plain fun, especially if you grew up playing the original game. Note: Its size requires a trailer truck to move it between venues. Right next to the "Mouse Trap" were the famed "Coke and Mentos" fountains. I can't believe these guys have made a living off of 140 bottles of spraying fizz.

HACKERSPACES

How many of you have wanted to start a project only to be foiled by not having the right tools or expertise to pull it off? Hackerspaces are groups of like-minded individuals who pool their resources and provide a community-operated place where members can come to teach, learn, and have access to tools to turn their dream projects into reality. Each facility is usually sustained through donations and yearly membership fees. A list of established hackerspaces is available through Wiki (see Resources).

Many hackerspace members exhibited samples of their ongoing projects at the Maker Faire. Among them was Andrew Lloyd Goodman, an electronic media artist from Providence, RI. Goodman demonstrated the "Hammer Project," an RGB LED array embedded into the head of a wooden sledgehammer. A user strikes a horizontal "bulletproof" glass surface, which can be scattered with items to destroy. An open-shuttered digital camera (on the other side of the glass) records the mayhem. Some results are available on Goodman's website (see Resources).

The 721st Mechanized Contest Battalion (MCB) is an amateur radio club from Warren County, NJ, that combines amateur (ham) radio with electronics, engineering, mechanics, building, and making. The club came to the Maker Faire to demonstrate its Emergency Antenna Platform System (E-APS) robot. The robot, which is designed for First Responder Organizations, will turn any parking lot lamppost into an instant antenna tower (see **Photo 2**).

PHOTO 2
This pole-climbing robot is easy to deploy at a moment's notice. There is no need for a ladder to get emergency communication antennas up high where they can be most effective.

PHOTO 3
Andrew Plumb showed me some unique ideas he was experimenting with using one of his 3-D printers. By printing the structural frame directly on tissue paper, ultra-light parts are practically ready to fly.

PHOTO 4
Rhode Island's 3D Printing Providence group displayed several 3-D figures. The figures were printed using some rather different materials, which gave the impression that the objects are made of wood or stone.

3-D PRINTING

Working by day as an analog/mixed-signal IC design engineer for Cortina Systems in Canada, Andrew Plumb needed a distraction. In the evenings, Plumb uses a MakerBot 3-D printer to create 3-D designs of plastic, like thousands of others experimenting with 3-D printing.

Plumb was not satisfied with simply printing plastic widgets. In fact, he showed me a few of his projects, which include printing plastic onto paper and cloth (see **Photo 3**). I guess that's what happens when you've been doing 3-D printing as long as Plumb has. His collection of printing devices began with an early MakerBot Cupcake (he has one of the first 3-D printers, serial number 0009).

Matt Stultz, from Rhode Island's 3D Printing Providence group, brought his experience in multi-material and advanced materials to the Maker Faire (see **Photo 4**). He gave a talk at the 3-D printing stage about some new materials, including high-impact polystyrene (HIPS); Laywood, which is a type of PLA embedded with wood fibers; and Laybrick, the PLA infused with chalk dust to emulate a stone texture.

3-D MISCELLANEOUS

By far, the majority of groups at the Maker Faire demonstrated some kind of 3-D printing. Others, while keeping with the 3-D theme, took different paths.

Tarrytown, NY-based Dave Seff is a Linux administrator by day, and a machinist once the sun goes down. Seff's Solar1 project uses 3-D printing and CNC machining technology to manufacture custom parts, gadgets, and items. His fully functioning CNC plasma cutting table was on display at the Maker Faire. A fire extinguisher is always at the ready in case sparks fly. Naturally, Seff machined all his own parts for the plasma cutter build and his 3-D printer, which he nicknamed "Steel Bug."

It was just a matter of time until someone introduced a personal scanner to create digital files of 3-D objects. The MakerBot Digitizer Desktop 3-D Scanner is the first I've seen (see **Photo 5**). It uses a laser, a turntable, and a CMOS camera to pick off 3-D points and output a STL file. The scanner will create a 3-D image from an object up to 8" in height and width. There is no third axis scanning, so you must plan your model's orientation to achieve the best results. Priced less than most 3-D printers, this will be a hot item for 3-D printing enthusiasts.

PHOTO 5
The MakerBot Digitizer Desktop 3-D Scanner is the first production scanner I've seen that will directly provide files compatible with the 3-D printing process. This is a long-awaited addition to MakerBot's line of 3-D printers. (Photo credit: Spencer Higgins)

OTHER INTERESTING STUFF

The Public Laboratory for Open Technology and Science (Public Lab) is a community that uses inexpensive DIY techniques to investigate "environmental concerns." For instance, the New York chapter featured two spectrometers, a you-fold-it cardboard version and a near-infrared USB camera-based kit. This community of educators, technologists, scientists, and community organizers believes they can promote action, intervention, and awareness through a participatory research model in which you can play a part.

Over the last few years, Arduino has become one of the fastest-growing microcontroller-based platforms for electronics enthusiasts. TinyCircuits thinks smaller and lighter is better. And so, the TinyDuino was born. This ultra-compact Arduino is 20 mm square (or round) with a miniature stacking connector that supports the typical expansion header signals (see **Photo 6**).

Tiny add-on shields offer Bluetooth, Wi-Fi, USB, sensors, display, and motor drives for your swarm of robots or airborne payload. The board-to-board stack-up height is only 3 mm.

Chris "The Carpenter" Robinson, runs Rocket Brand Studios from Cape Cod, MA. His goal is to provide educators with an inexpensive robotic platform that can be used to teach a basic robotic class. "I spent several months designing and perfecting the 'Tadpole' based on the requests I heard: Simple, non-solder, inexpensive, and with room to expand capabilities in the future," Robinson explained.

A basic non-microcontroller platform costs less than $30.

Jackson, WY, is home to Cultivar, a company that is developing and manufacturing RainCloud, a web-connected irrigation system and application platform. RainCloud links mobile devices to lawns, gardens, and plants by combining automated water valves, soil sensors, a Wi-Fi-enabled programmable computer, and custom web applications.

Like many startups, Cultivar is using Kickstarter to test the waters for this project, which includes a latching DC solenoid valve, a Decagon Devices EC-5 soil moisture probe, and a Raspberry Pi microcontroller (see Resources).

Plantation, FL-based TeraBatt uses hybrid battery technology to store energy collected in the most efficient way possible. For instance, a home generator is designed for maximum efficiency at one output, anything less and you're not running at peak efficiency. TeraBatt's technology draws a constant output from the generator, enabling hybrid batteries to charge at the generator's peak efficiency. Your house then uses an inverter to draw from these batteries, which maximizes this conversion efficiency.

Mixed-media maker and blogger Sarah Hodsdon (*Sarahndipitous*) was at the Maker Faire to engage attendees not with flashy electronics, but low-tech items already found in most homes. While her purpose is to foster an environment that will produce the "next generation" of makers one kitchen table at a time, it was our conversation that hit big with me. Our discussion included the "real cost of doing business," which is an important change in attitude that is sure to become an issue as we move beyond the oil age. I urge you to read her blog entry "The Cost of Business" (see Resources).

Asheville, NC-based Beatty Robotics is not your average robotics company. The Beatty team is a family that likes to share fun robotic projects with friends, family, and other roboticists around the world. The team consists of Dad (Robert) and daughters Camille "Lunamoth" and Genevieve "Julajay." The girls have been mentored in electronics, software programming, and workshop machining. They do some unbelievable work (see **Photo 7**). Everyone has a hand in designing, building, and programming their fleet of robots. The Hall of Science is home to one of their robots, the Mars Rover.

HACKER COMMUNITIES

As I alluded to earlier, while hackerspaces are a new concept to many, they have been in existence for years. I remember when the media coined the term "hacker" as someone who illegally accessed a computer system by circumventing its security. Early arrests for this crime often resulted in hiring the perpetrator to take advantage of their expertise and improve security. While hacking can be used for evil purposes, most hackers just wanted to show off their talents. Thus the term is embraced by many to mean "a cleverness or skill in altering some equipment's intended purpose."

Today, "hacker" means "one with inexperience or unskilled." This defines the essence of a hackerspace or community workshop: A place where one can get experience and learn the skills necessary to complete a dream; a new kind of school. There are no sheep skins, just people seeking personal satisfaction.

Some organizations have taken the concept one step further. California-based TechShop began spreading its wings across America

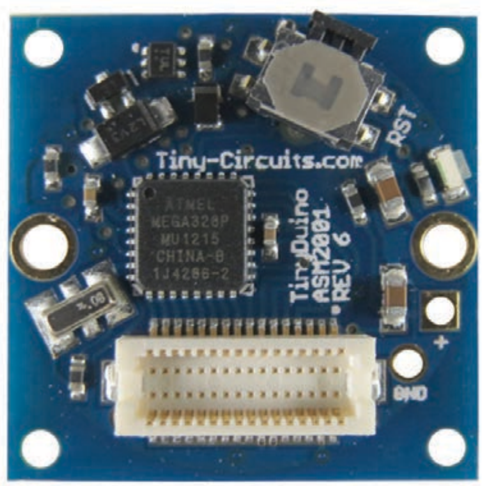


PHOTO 6
Designed with size and weight in mind, TinyCircuits's TinyDuino board is about the size of a quarter. It uses a high-density connector to enable the addition of compatible mini shield boards.

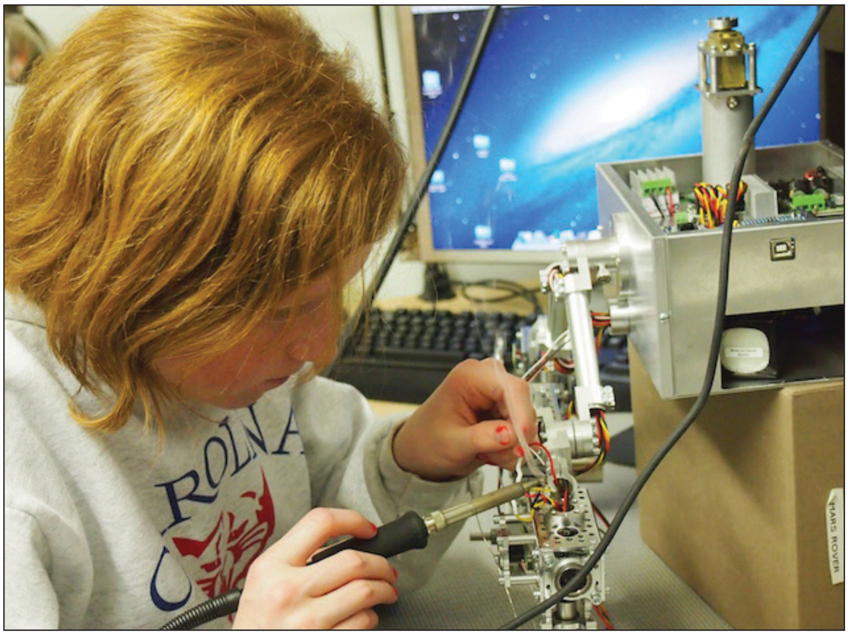


PHOTO 7
Beatty Robotics is a family of makers that produces some incredible models. Young Camille Beatty handles the soldering, but is also well-versed in machining and other areas of expertise.

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@imaginethatnow.com or at www.imaginethatnow.com.

in 2006. Each facility includes laser cutters; plastics and electronics labs; machine, wood, and metal working shops; a textiles department; welding stations; and a waterjet cutter. Members have open access to design software, including the entire Autodesk Design Suite. Project areas with large worktables are available for completing projects and collaborating with others.

Under the TechShop name, anyone can participate in two areas: classes and membership. Classes are offered to members and non-members on a fee-per-class basis. You can become an annual or a monthly member. Membership enables you to schedule full use of any equipment in blocks of time, day or night. Other areas may include classrooms, offices, lounges, storage areas, and a storefront.

If you looked at the hackerspace website I mentioned earlier, you may have noticed there are plenty of independent spaces established other than the TechShop group. Most of these have originated via local meet-ups and other gatherings of those who get together to share their interests in a particular area (e.g., art, cars, photography, sports, recreation, technology, etc.). The groups may have started small, but grew as their popularity increased. If you're a designer, you know how expensive it can be to purchase the right tool for the job. These spaces have become local libraries for tools. In fact, I can foresee your local library being forced to morph into new covenants, or find itself unfunded as cities pare down their budgets.

What do Kickstarter, Indiegogo, and RocketHub have in common with community spaces? It's a new way of thinking. Crowdfunding puts you in front of people who want to buy your product, connect with you, and team up with you in many ways. Community and collaboration are at the heart of it. My town's local library is beginning to offer a variety of free classes. This is a great place to start, make connections, and form alliances.

CIRCUIT CELLAR CONNECTION

Many of you have been with *Circuit Cellar* since Steve Ciarcia started publishing in 1987. From that first issue, you knew what to expect: technical articles and projects by and for engineers, entrepreneurs, and hobbyists. As a world leader in electronic publications, Elektor International Media (EIM) recognized *Circuit Cellar* as the established engineering force in the US. A marriage brought *Circuit Cellar*'s gift of sharing technology to Elektor's world presence. This assured not only a continuation of the intensive, exploratory articles about hardware and software methods for embedded-control systems *Circuit Cellar* readers have enjoyed for years, but also broadened its outreach across oceans.

EIM is known for its passion for electronics. It has expanded its publishing services by offering an in-house design laboratory and PCB design department. I'd say EIM is taking steps to create a worldwide community of sorts, a place where like-minded entrepreneurs can go to bring their dreams to fruition. We are all makers. If we have access to the education and tools we need, there is no stopping our creativity. This is the stuff the future is made of.

PROJECT FILES

circuitcellar.com/ccmaterials

RESOURCES

3D Printing Providence, www.3dppvd.org.

721st Mechanized Contest Battalion (MCB), www.wc2fd.com.

Beatty Robotics, http://beatty-robotics.com.

ClothBot Designs, www.clothbot.com.

Elektor, "Elektor PCB Service."

Andrew Lloyd Goodman, "LightSmithing," http://andrewlloydgoodman.com.

Cultivar, Inc., "Cultivar's RainCloud: Control Your Water Intelligently," www.kickstarter.com/projects/1440288384/cultivars-raincloud-control-your-water-intelligent.

Maker Faire, www.makerfaire.com.

New York Hall of Science, www.nysci.org.

The Public Laboratory for Open Technology and Science (Public Lab), www.publiclab.org.

Rocket Brand Studios, http://rocketbrandstudios.com.

Sarahndipitous, sarahndipitousdesigns.com.

Solar1 Labs, Inc., www.solar1.net.

TerraBatt, www.terabatt.com.

Wiki, "List of Hacker Spaces," http://hackerspaces.org.

SOURCES

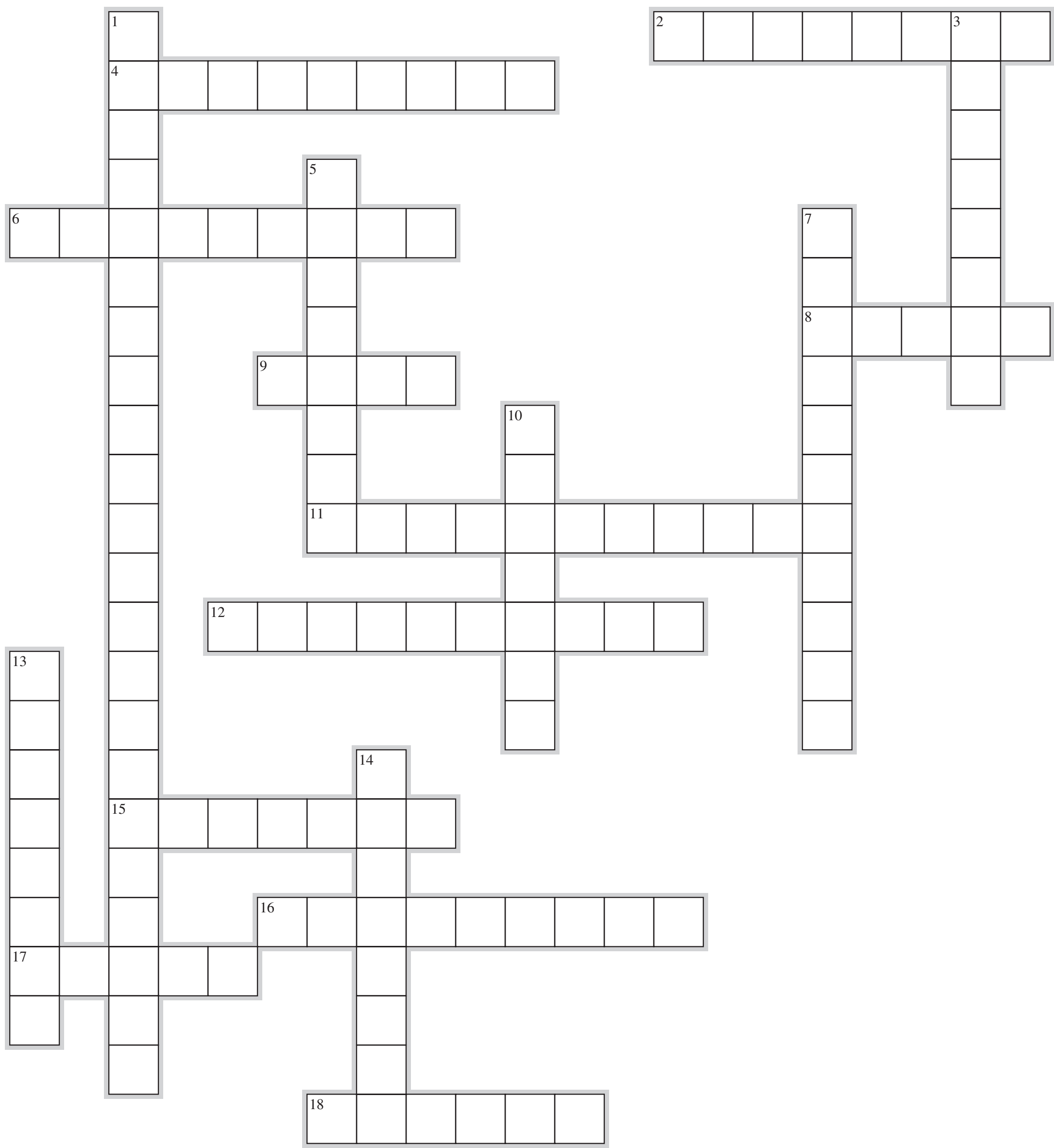
Autodesk Design Suite
Autodesk, Inc. | www.autodesk.com

Digitizer Desktop 3-D Scanner
MakerBot Industries | www.makerbot.com

TinyDuino board
TinyCircuits | www.tinycircuits.com

CROSSWORD FEBRUARY 2014

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



ACROSS

2. Separately insulated strands woven together [two words]
4. First in a message buffer's line [two words]
6. Measures a processor's floating-point unit performance
8. Negatively charged atom
9. Used to manipulate mathematical logic
11. i.e., franklin (Fr)
12. Typically requires a soundfield microphone
15. This device can send data between networks and it can forward data to individual systems in a network
16. This CRT technology was originally introduced in the 1960s
17. The "O" in CMOS
18. Used to prevent or stop something from spinning

DOWN

1. X_E [two words]
3. A barrier-grid storage tube
5. This oscillator uses two-terminal electrical components to create a specific oscillation frequency
7. Texas Instruments's open-source SBC
10. A network that is created using a wireless Bluetooth connection
13. Designed to accelerate electrons
14. More than 1,000,000,000,000 bytes

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

TEST YOUR EQ

Contributed by David Tweed

ANSWER 1

You get equations in this form when you do mesh analysis of a circuit. Each equation represents the sum of the voltages around one loop in the mesh.

ANSWER 2

The coefficients on the left side of each equation represent resistances. Resistance multiplied by current (the unknown Xa, Xb, and Xc values) yields voltage.

The "bare" numbers on the right side of each equation represent voltages directly (i.e., independent voltage sources).

ANSWER 3

To solve the equations directly, start by solving the third equation for Xc and substituting it into the other two equations:

$$Xc = 1/2\ Xa + 1/2\ Xb + 1/2$$

$$21Xa - 10Xb - 5Xa - 5Xb - 5 = 1$$
$$-10Xa + 22Xb - 5Xa - 5Xb - 5 = -2$$

$$16Xa - 15Xb = 6$$
$$-15Xa + 17Xb = 3$$

Solve for Xa by multiplying the first equation by 17 and the second equation by 15 and then adding them:

$$272Xa - 255Xb = 102$$
$$-225Xa + 255Xb = 45$$

$$47Xa = 147 \rightarrow Xa = 147/47$$

Solve for Xb by multiplying the first equation by 15 and the second equation by 16 and then adding them:

$$240Xa - 225Xb = 90$$
$$-240Xa + 272Xb = 48$$

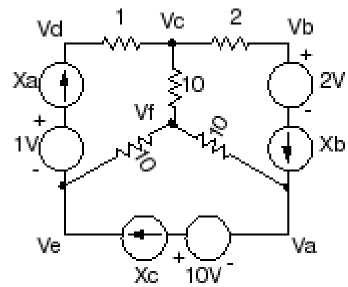
$$47Xb = 138 \rightarrow Xb = 138/47$$

Finally, substitute those two results into the equation for Xc:

$$Xc = 147/94 + 138/94 + 47/94 = 332/94 = 166/47$$

ANSWER 4

The circuit is a mesh comprising three loops, each with a voltage source. The common elements of the three loops are the three 10-Ω resistors, connected in a Y configuration (see the figure below).



The values of the voltage sources in each loop are given directly by the equations, as shown. To verify the numeric solution calculated previously, you can calculate all of the node voltages around the outer loop, plus the voltage at the center of the Y, and ensure they're self-consistent.

We'll start by naming Va as ground, or 0 V:

$$Vb = Va + 2\ V = 2\ V$$

$$Vc = Vb + 2\ \Omega \times Xb = 2V + 2\ \Omega \times 138/47\ A = 370/47\ V = 7.87234\ V$$

$$Vd = Vc + 1\ \Omega \times Xa = 370/47\ V + 1\ \Omega \times 147/47A = 517/47\ V = 11.000\ V$$

$$Ve = Vd - 1\ V = 11.000\ V - 1.000\ V = 10.000\ V$$

$$Va = Ve - 10\ V = 0\ V$$

which is where we started.

The center node, Vf, should be at the average of the three voltages Va, Vc, and Ve:

$$0\ V + 370/47\ V + 10\ V/3 = 840/141\ V = 5.95745\ V$$

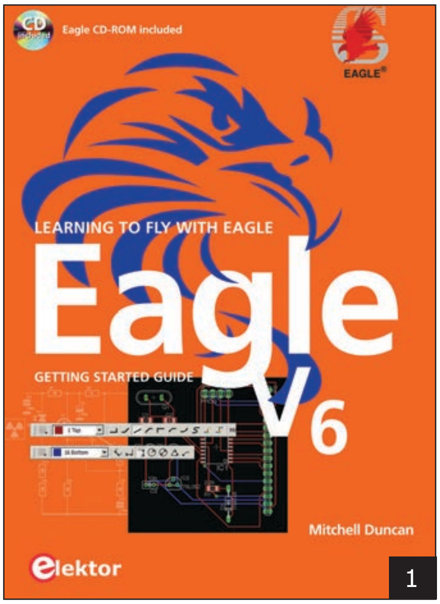
We should also be able to get this value by calculating the voltage drops across each of the three 10-Ω resistors:

$$Va + (Xc - Xb) \times 10\ \Omega = 0\ V + (166 - 138)/47A \times 10\ \Omega = 280/47\ V = 5.95745\ V$$

$$Vc + (Xb - Xa) \times 10\ \Omega = 370/47V + (138\text{-}147)/47A \times 10\ \Omega = 280/47\ V = 5.95745\ V$$

$$Ve + (Xa - Xc) \times 10\ \Omega = 10\ V + (147\text{-}166)/47\ A \times 10\ \Omega = 280/47\ V = 5.95745\ V$$

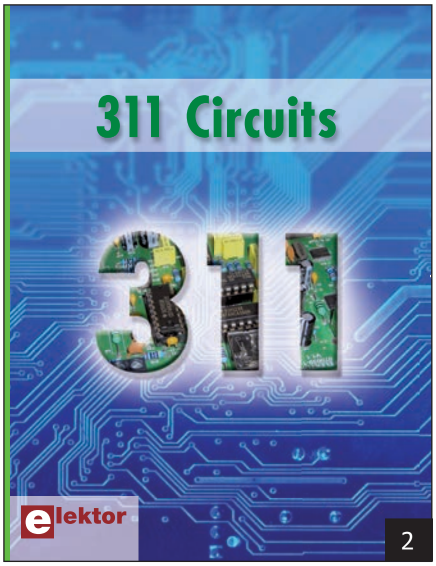
CC SHOP



1 LEARNING TO FLY WITH EAGLE V6

EAGLE is a user-friendly, powerful, and affordable software package for efficient PCB design. It can be used on most main computing platforms including: Microsoft Windows, Linux, and Apple Mac OS X. *EAGLE V6* will benefit novices and professionals who are eager to learn about EAGLE or may be migrating from another CAD package. From schematic and layout editing tools to project completion, this book will help you achieve your PCB fabrication goals.

Item #: BK-ELNL-978-1-907920-20-2



2 311 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.

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

3 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 seven editions.

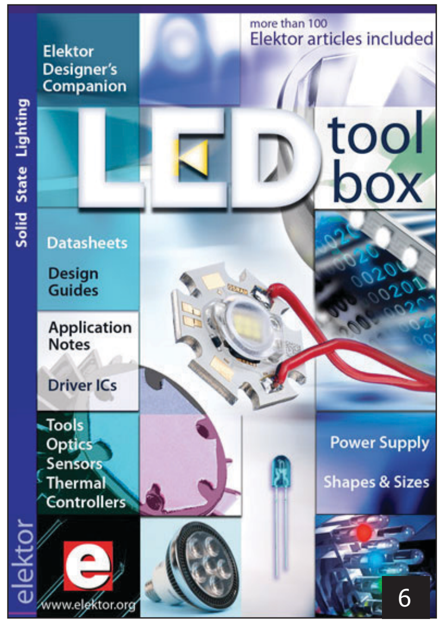
4 ADUC841 MICROCONTROLLER DESIGN MANUAL

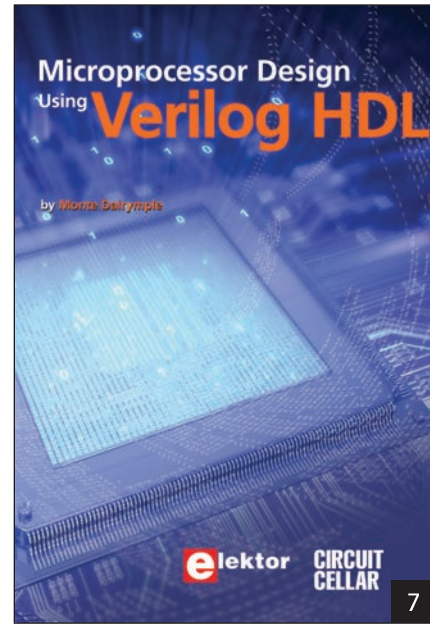
This manual presents a comprehensive guide to designing and programming with the Analog Devices ADuC841 microcontroller and other microcontrollers in the 8051 family. It includes a set of introductory labs that detail how to use these microcontrollers' most standard features, and includes a set of more 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 µVision4 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
$35

CC SHOP





5 CIRCUIT CELLAR ISSUE PDFS

Have you ever missed an issue of *Circuit Cellar* and didn't know where to find it? Grab a digital version! From 1998 to the present, dive into more than 100 issues of embedded electronics insights that you may have missed. Can you envision the evolution and development across a decade? With technology advancing every day, don't fall behind. Learn about wearable wireless transceivers from Mathew Laibowitz and Joseph Paradiso or a DIY SBC from Oscar Vermeulen and Andrew Lynch. The topics and projects are endless!

6 LED TOOLBOX

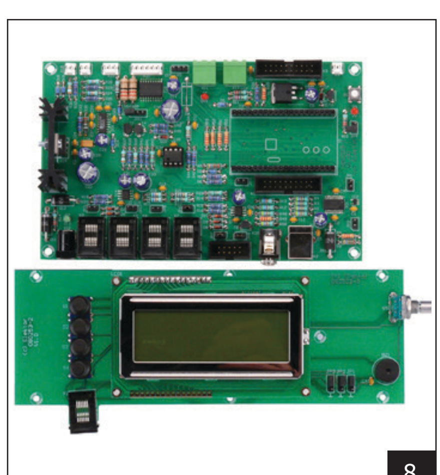
A toolbox is typically used to store a variety of useful items. The LED Toolbox functions in the same all-inclusive fashion. This DVD-ROM contains technical documentation, including electrical characteristics and mounting information, as well as manufacturer datasheets and design guides. In addition to optical systems, the LED Toolbox also offers several hundred drivers for powering and controlling LEDs in different configurations, including boost, charge pump, and constant current. This DVD is accessible through PDF files and easy browsing using an HTML menu.

Item #: DVD-ELNL-978-90-5381-245-7

7 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



8 AUTOMATIC RUNNING-IN BENCH

Referenced in *Elektor* April 2009, this kit consists of a motherboard and a portable terminal, which is connected by a cable to a six-pin RJ11 socket on the motherboard. The automatic running-in bench kit is designed so your project can be built primarily with conventional devices. This unique kit can be used for several things, specifically standard-model servomotor throttle control, configurable travel and direction of movement, mixture adjustment managed by the on-board software, emergency-stop push button, and USB linking.

Item #: EPS-ELNL-080253-71

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@smmarketing.us.

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

Can anyone deny that we're on the verge of major breakthroughs in the fields of embedded development, microcomputing, wireless communications, and robot design? Tech the Future is a section devoted to the ideas and stories of innovators who are developing the groundbreaking technologies of tomorrow.

The Future of Inkjet-Printed Electronics

By Benjamin Cook

Benjamin Cook is a PhD candidate in the School of Electrical and Computer Engineering at the Georgia Institute of Technology. He works with the ATHENA research group, which is headed by Professor Manos M. Tentzeris. Benjamin's research focuses on the development and application of 2-D and 3-D vertically integrated inkjet fabrication technologies for RF and sub-terahertz wireless applications. He is currently working with the industry to bring this technology to the consumer and mass manufacturing markets.

Over the past decade, major advances in additive printing technologies in the 2-D and 3-D electronics fabrication space have accelerated additive processing—printing in particular—into the mainstream for the fabrication of low-cost, conformal, and environmentally friendly electronic components and systems. Printed electronics technology is opening an entirely new world of simple and rapid fabrication to hobbyists, research labs, and even commercial electronics manufacturers.

Historically, PCBs and ICs have been fabricated using subtractive processing techniques such as photolithography and mechanical milling. These traditional techniques are costly and time-consuming. They produce large amounts of material and chemical waste and they are also difficult to perform on a small scale for rapid prototyping and experimentation.

To overcome the limitations of subtractive fabrication, the ATHENA group at the Georgia Institute of Technology (Georgia Tech) has been developing an innovative inkjet-printing platform over the past decade that can print complex, vertical ICs directly from a desktop inkjet printer.

To convert a standard desktop inkjet printer into an electronics fabrication platform, custom electronic inks developed by Georgia Tech replace the standard photo inks that are ejected out of the printer's piezoelectric nozzles. Inks for depositing conductors, insulators/dielectrics, and sensors have all been developed. These inks can print not only single-layer flexible PCBs, but they can also print complex, vertically integrated electronic structures (e.g., multilayer wiring with interlayer vias, parallel-plate capacitors, batteries, and sensing topologies to sense gas, temperature, humidity, and touch).

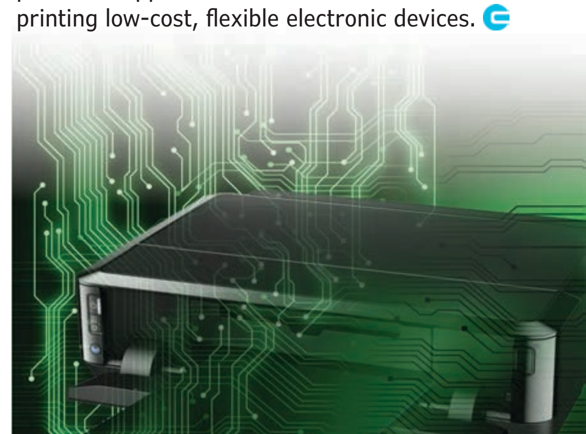
To create highly efficient electronic inks, which are the key to the printing platform, Georgia Tech researchers exploit the nanoscale properties of electronic materials. Highly conductive metals (e.g., gold, silver, and copper) have very high melting temperatures of approximately 1,000°C when the materials are in their bulk or large-scale form. However, when these metals are decreased to nanometer-sized particles, their melting temperature dramatically decreases to below 100°C. These nanoscale particles can then be dispersed within a solvent (e.g., water or alcohol) and printed through an inkjet nozzle, which is large enough to pass the nanoparticles. After printing, the metal layer printed with nanoparticles is heated at a low temperature, which melts the particles back into a highly conductive metal to produce very low-resistance electrical structures.

Utilizing nanomaterials has enabled the creation of plastic, ceramic, piezoelectric, and carbon nanotube and graphene inks, which are the fundamental building blocks of a fully printed electronics platform. The inks are then tuned to have the correct viscosity and surface tension for a typical desktop inkjet printer.

By loading these nanomaterial-based conductive, dielectric, and sensing inks into the different-colored cartridges of a desktop inkjet printer, 3-D electronics topologies such as metal-insulator-metal (MIM) capacitors can then be created by printing the different inks on top of each other in a layer-by-layer deposition. Since printing is a non-contact additive deposition method, and the processing temperatures are below 100°C, these inks can be printed onto virtually any substrate, including standard photo paper, plastic, fabrics, and even silicon wafers to interface with standard ICs with printed feature sizes below 20 µm.

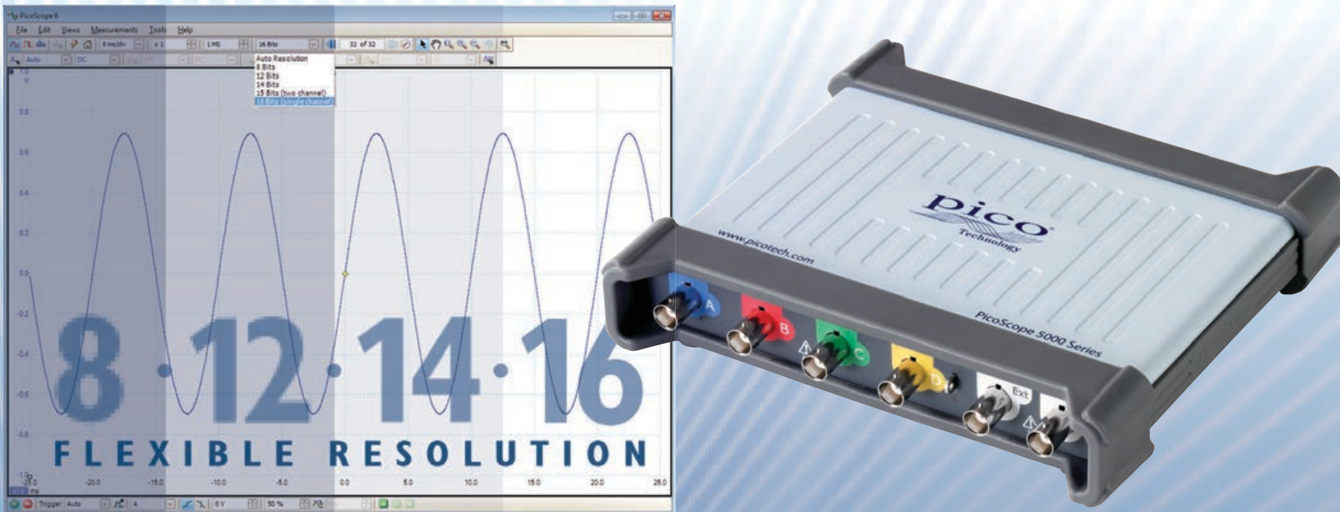
The Georgia Tech-developed printing platform is a major breakthrough. It makes the cost of additively fabricating circuits nearly the same as printing a photo on a home desktop inkjet printer—and with the same level of simplicity and accessibility.

These advancements in 2-D electronics printing combined with current research in low-cost 3-D printing are enabling commercial-grade fabrication of devices that typically required clean room environments and expensive manufacturing equipment. Such technology, when made accessible to the masses, has the potential to completely change the way we think about building, interacting with, and even purchasing electronics that can be digitally transmitted and printed. While the printing technology is currently at a mature stage, we have only scratched the surface of potential applications that can benefit from printing low-cost, flexible electronic devices.



pico Technology

WHY COMPROMISE
SPEED v ACCURACY?
HAVE IT ALL

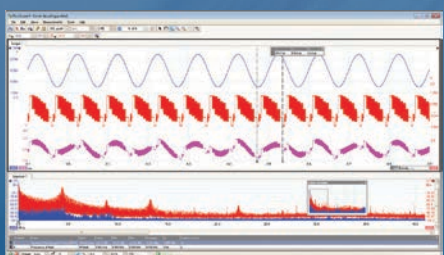


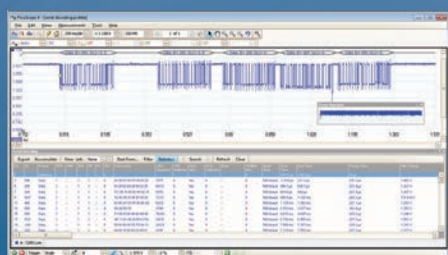
PicoScope	PicoScope 5442A	PicoScope 5442B	PicoScope 5443A	PicoScope 5443B	PicoScope 5444A	PicoScope 5444B
Channels	4					
Bandwidth	All modes: 60 MHz		8 to 15-bit modes: 100 MHz 16-bit mode: 60 MHz		8 to 15-bit modes: 200 MHz 16-bit mode: 60 MHz	
Sampling rate - ETS (8-bit mode only)	2.5 GS/s		5 GS/s		10 GS/s	
Buffer memory (8-bit) *	16 MS	32 MS	64 MS	128 MS	256 MS	512 MS
Buffer memory (≥ 12-bit)*	8 MS	16 MS	32 MS	64 MS	128 MS	256 MS
Resolution (enhanced)**	8 bits, 12 bits, 14 bits, 15 bits, 16 bits (hardware resolution + 4 bits)					
Signal Generator	Function generator	AWG	Function generator	AWG	Function generator	AWG

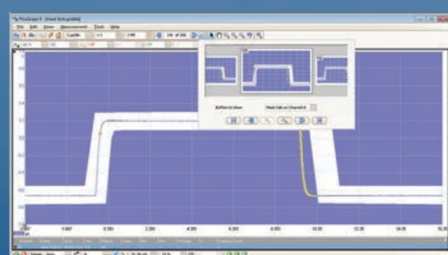
Max. sampling rate	8-bit mode	12-bit mode	14-bit mode	15-bit mode	16-bit mode
Any 1 channel	1 GS/s	500 MS/s	125 MS/s	125 MS/s	62.5 MS/s
Any 2 channels	500 MS/s	250 MS/s	125 MS/s	125 MS/s	-
Any 3 channels	250 MS/s	125 MS/s	125 MS/s	-	-
Four channels	250 MS/s	125 MS/s	125 MS/s	-	-

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.

FLEXIBLE RESOLUTION OSCILLOSCOPE







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/PCO516 CALL TOLL FREE: 1-800-591-2796


Power of the Quoting Revolution
in the palm of your hand!
At a click of a button we will find the best
price in the market compiled in one easy to
read spreadsheet
...just like your BOM.
Join the Revolution
Happening Now @
www.PCBnet.com
847-806-0003 sales@PCBnet.com
ITAR, I SO 9001:2008, UL Approved